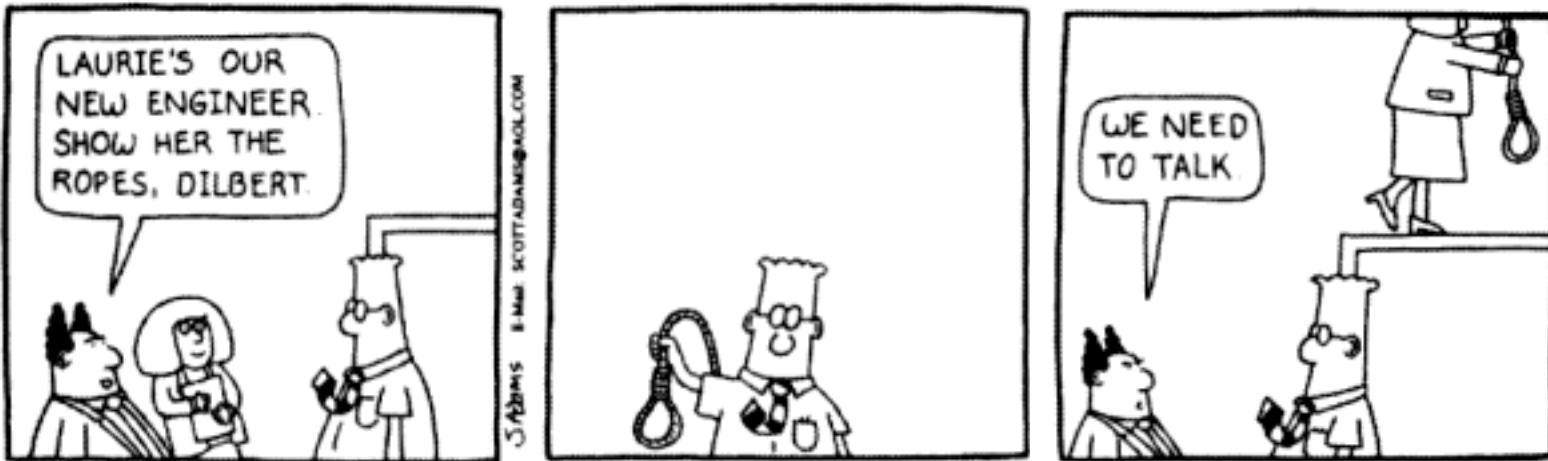


LIS tutorial

Sujay Kumar



Outline



Introduction



Building LIS, using repository, code organization



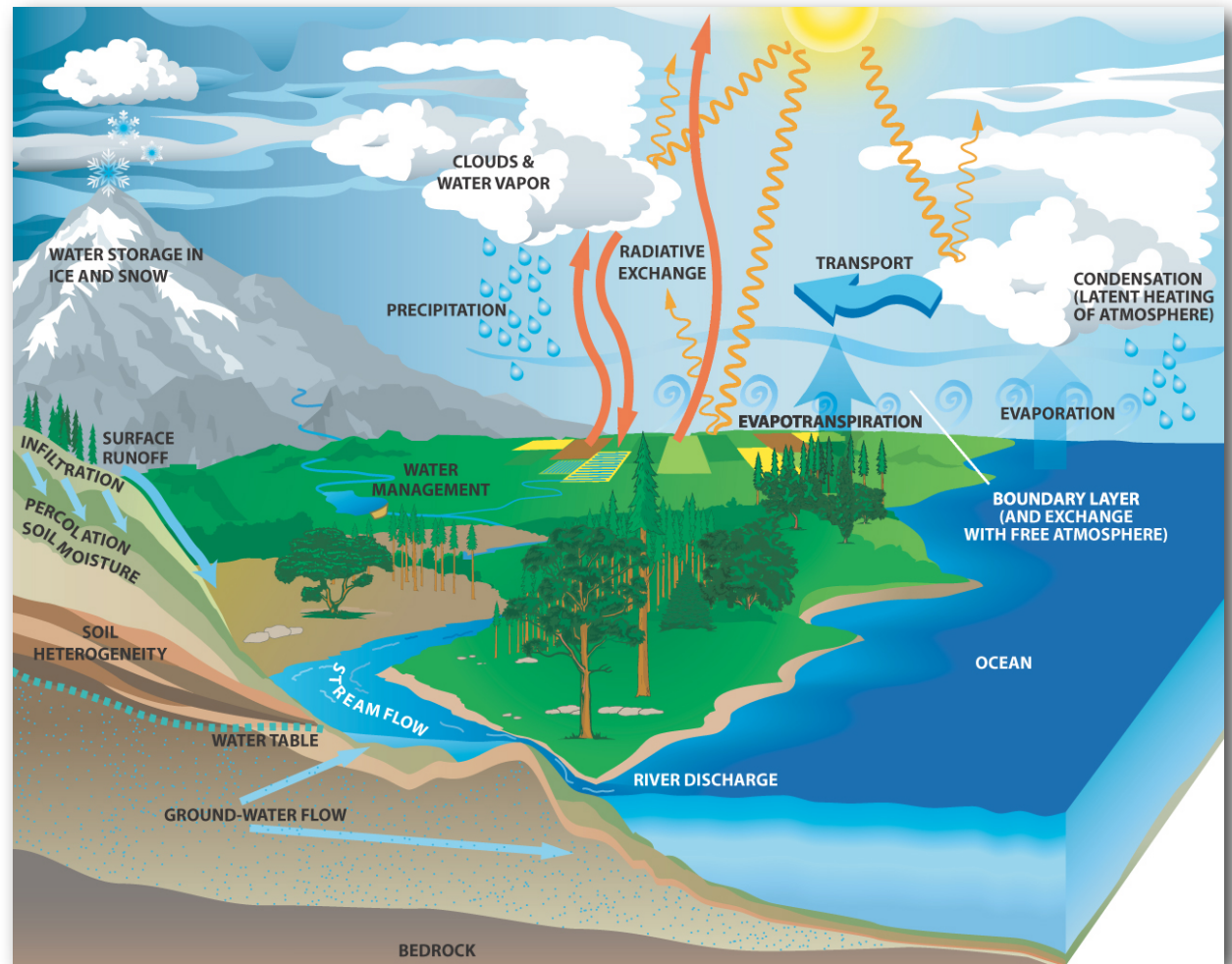
Software design philosophy, architecture



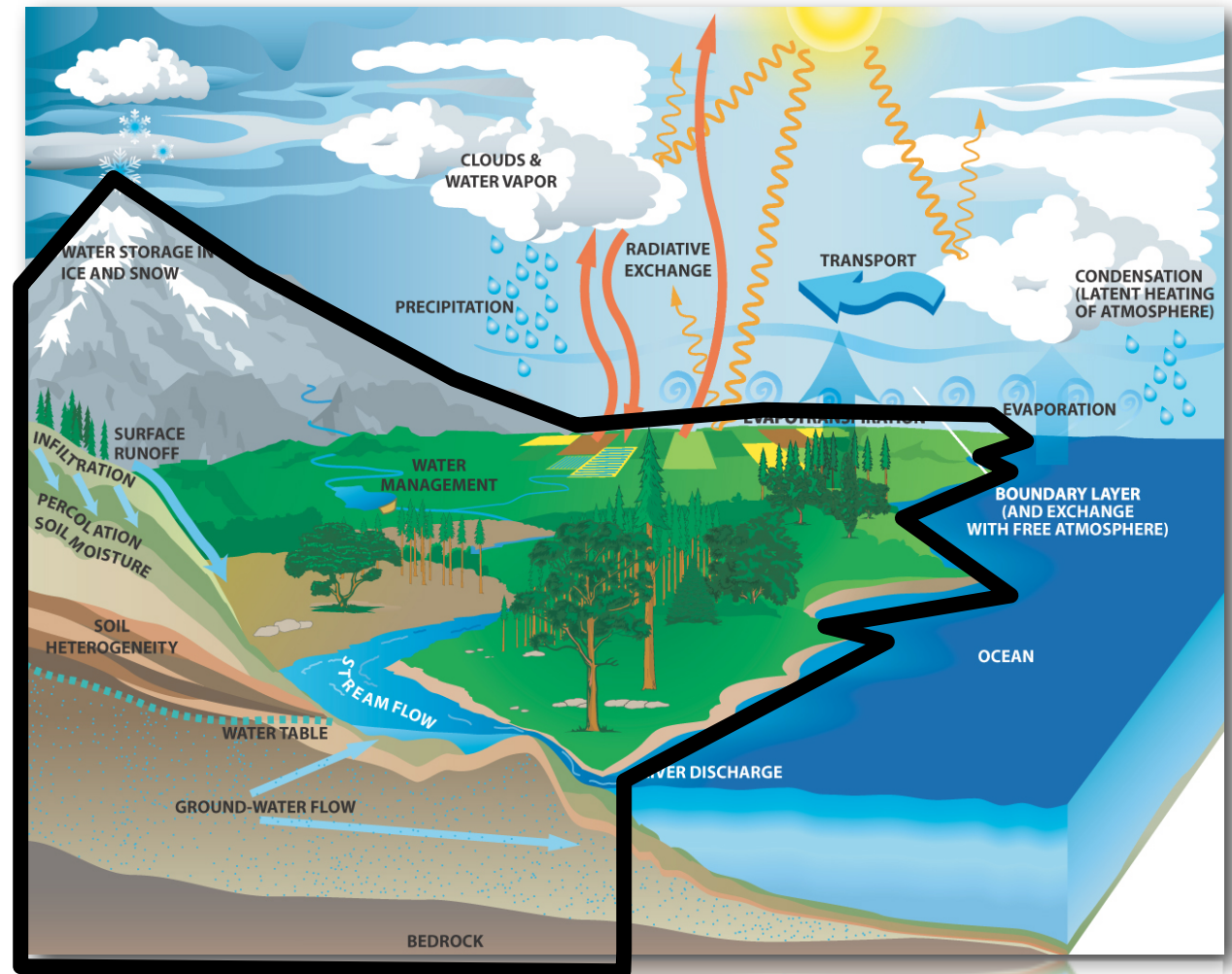
Customizing LIS



Visualizing LIS output

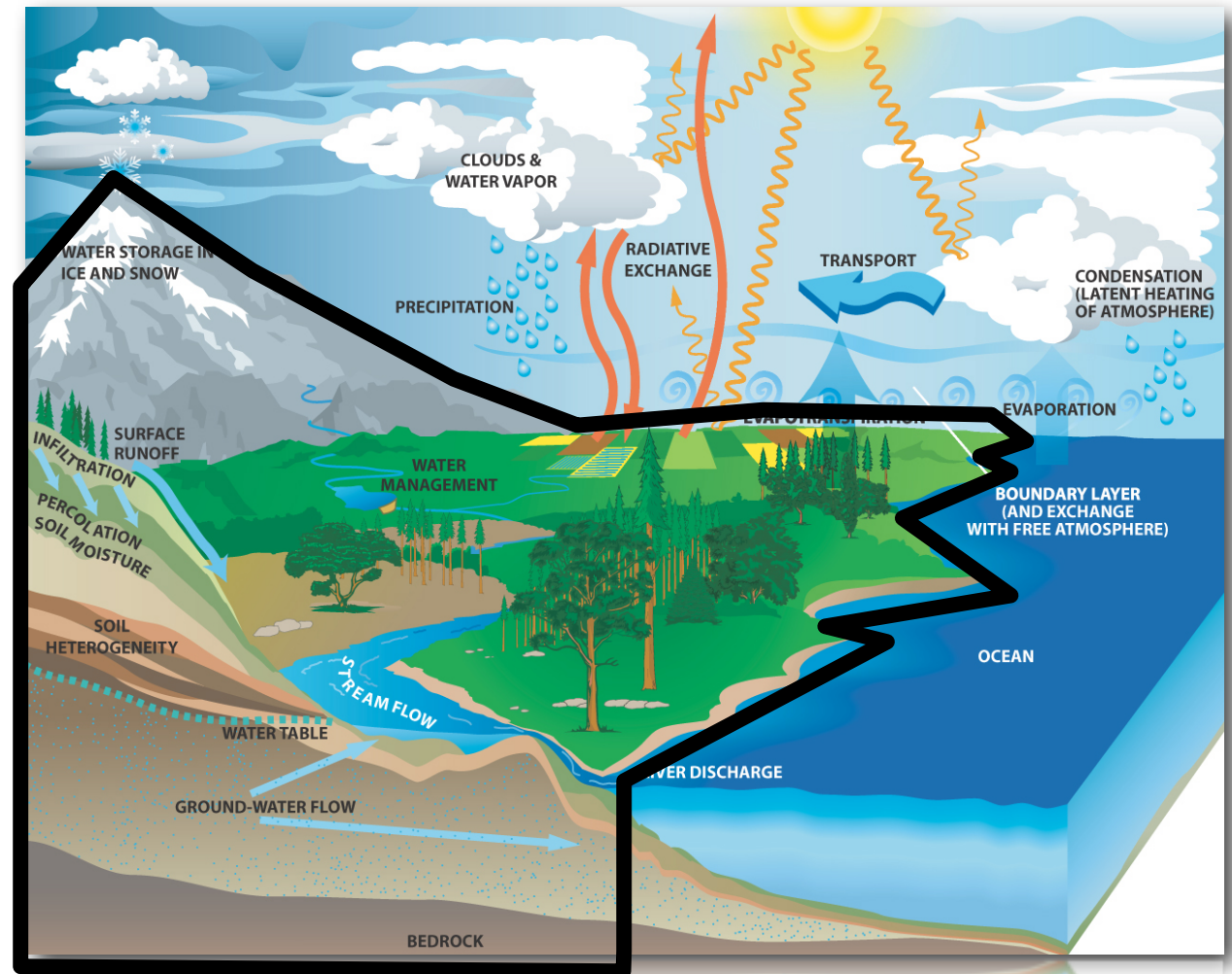


What is LIS?



What is LIS?

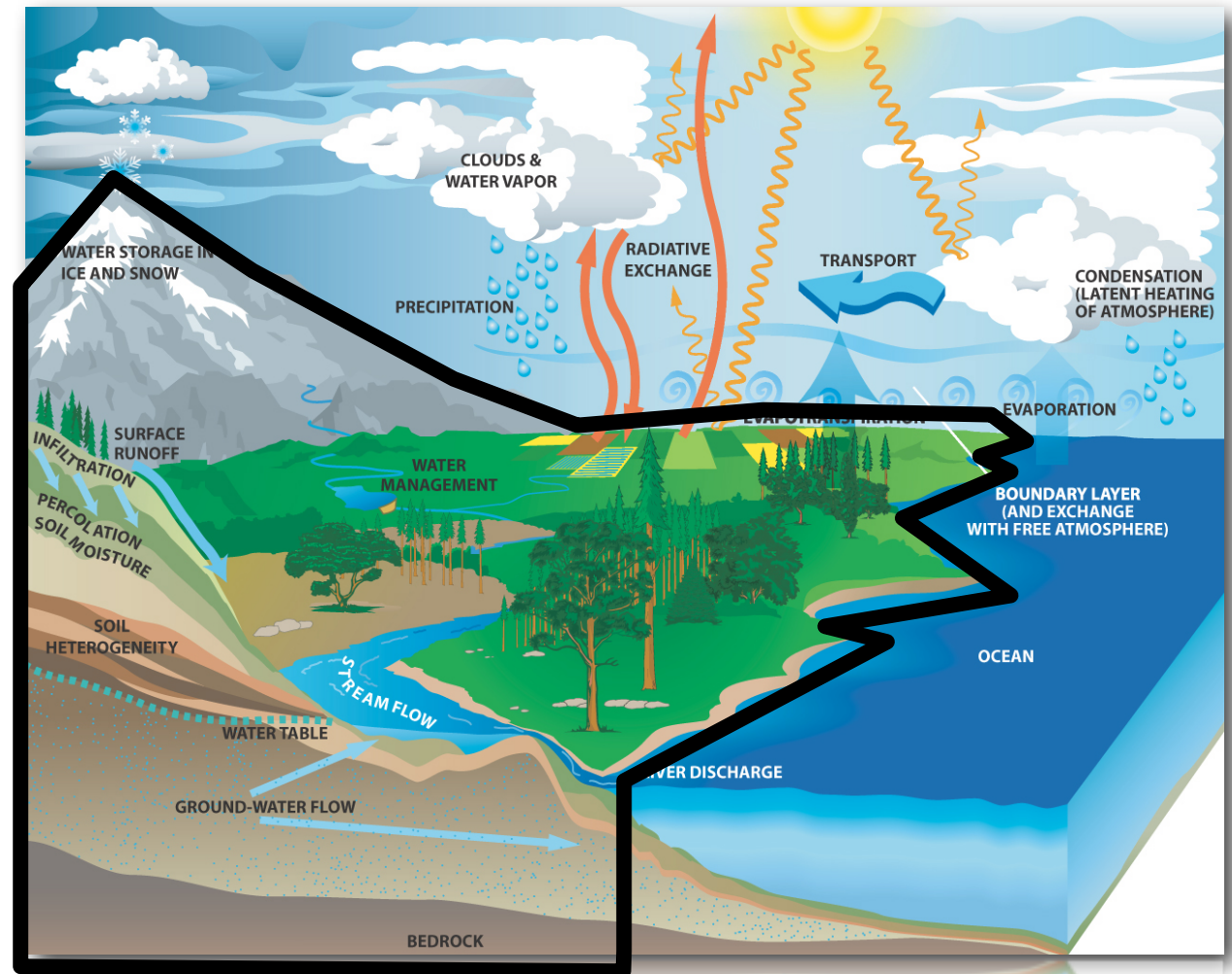
A system to study land surface processes and land-atmosphere interactions



What is LIS?

A system to study land surface processes and land-atmosphere interactions

“Use best available observations” to force and constrain the models

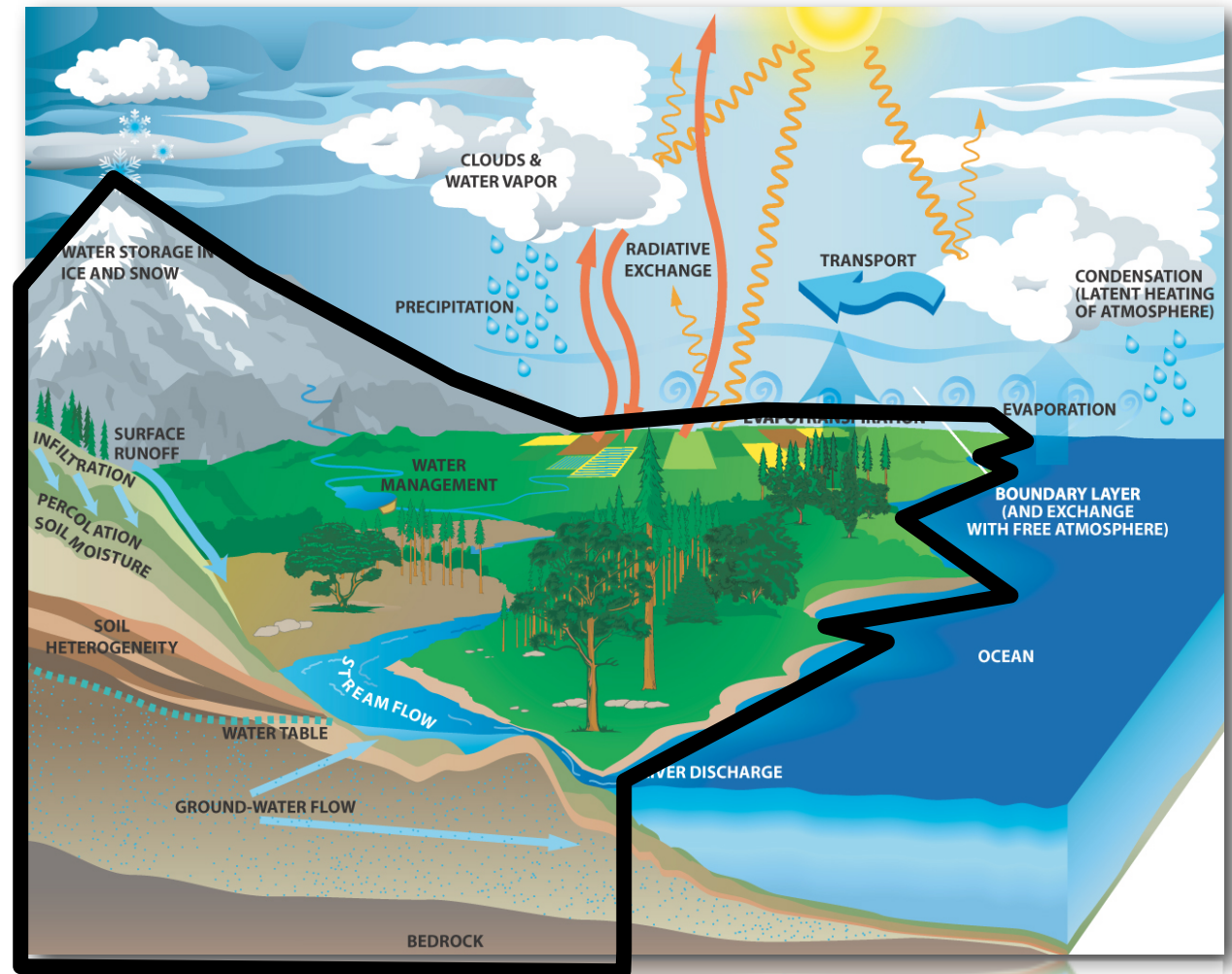


What is LIS?

A system to study land surface processes and land-atmosphere interactions

“Use best available observations” to force and constrain the models

Applications: Weather and climate model initialization, water resources management, natural hazards management



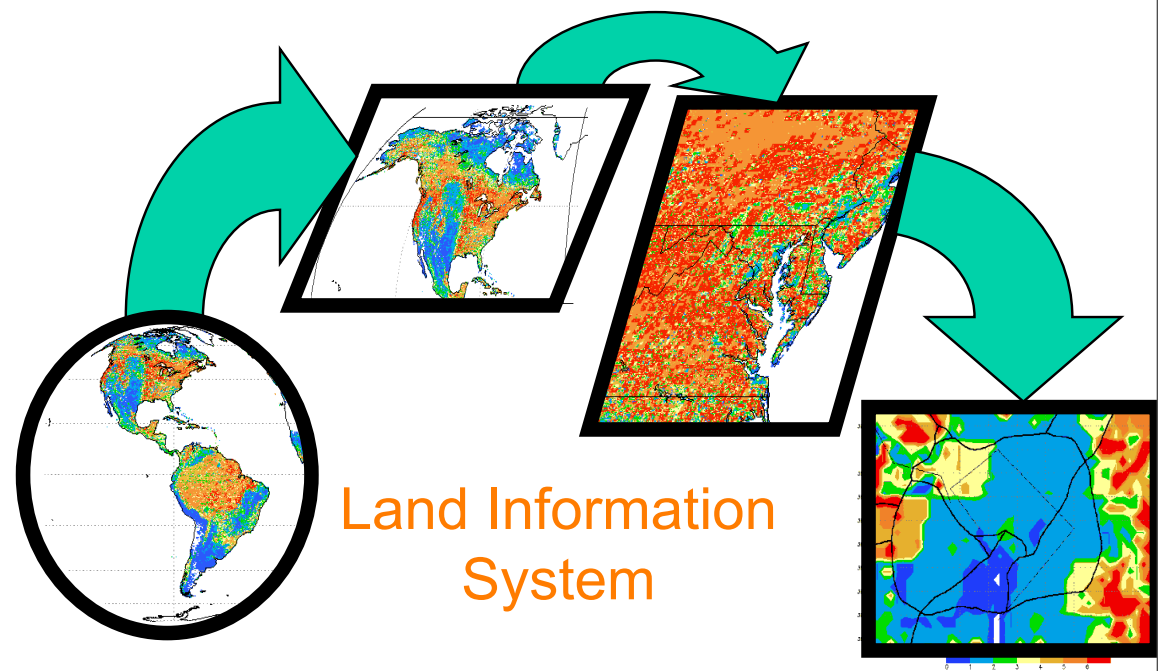
What is LIS?

- Need a system viable at different spatial and temporal scales

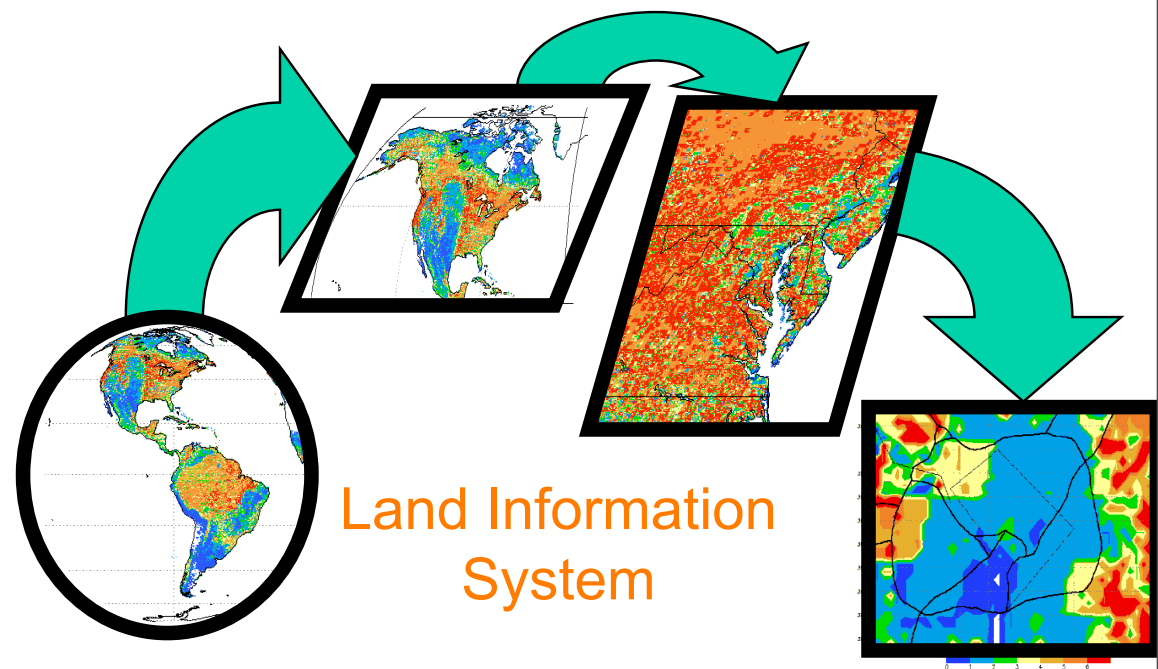
- ☐ Need a system viable at different spatial and temporal scales
- ☐ Be able to demonstrate the impact of observations at the scale of observations themselves

- ☐ Need a system viable at different spatial and temporal scales
- ☐ Be able to demonstrate the impact of observations at the scale of observations themselves
- ☐ Explicit characterization of the land surface at the same spatial scales as that of cloud and precipitation processes helps in improving the characterization of land-atmosphere interactions

- ❑ Need a system viable at different spatial and temporal scales
- ❑ Be able to demonstrate the impact of observations at the scale of observations themselves
- ❑ Explicit characterization of the land surface at the same spatial scales as that of cloud and precipitation processes helps in improving the characterization of land-atmosphere interactions



- ❑ Need a system viable at different spatial and temporal scales
- ❑ Be able to demonstrate the impact of observations at the scale of observations themselves
- ❑ Explicit characterization of the land surface at the same spatial scales as that of cloud and precipitation processes helps in improving the characterization of land-atmosphere interactions
- ❑ Need scalable, high performance computing support to deal with computational challenges



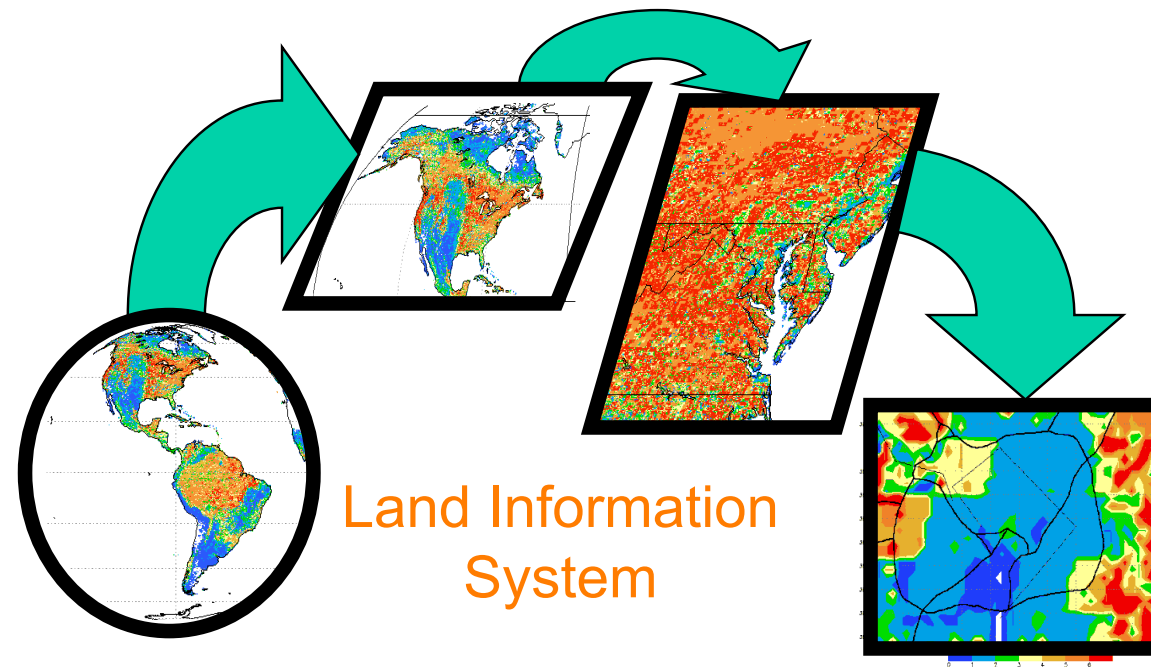
☐ Need a system viable at different spatial and temporal scales

☐ Be able to demonstrate the impact of observations at the scale of observations themselves



☐ Explicit characterization of the land surface at the same spatial scales as that of cloud and precipitation processes helps in improving the characterization of land-atmosphere interactions

☐ Need scalable, high performance computing support to deal with computational challenges

☐ Need advanced land surface models and modeling tools (data assimilation, optimization, uncertainty modeling)



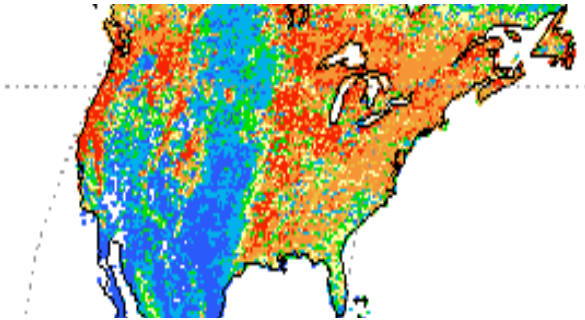
LIS - heritage

-  LIS is a land surface modeling and data assimilation system (LDAS)
-  Capable of modeling at different spatial scales, globally and regionally

Kumar et al. (2006): Land Information System: An interoperable Framework for High Resolution Land Surface Modeling, Environmental Modeling and Software, Vol 21, pp 1402-1415.

LIS - heritage

- LIS is a land surface modeling and data assimilation system (LDAS)
- Capable of modeling at different spatial scales, globally and regionally

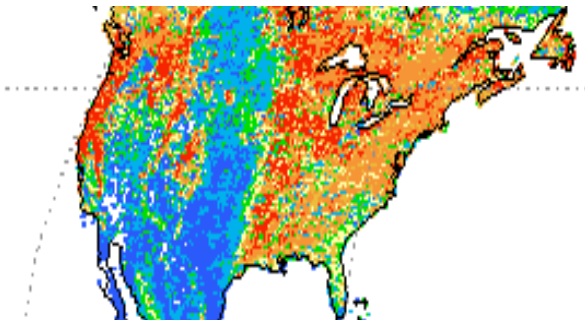


North American LDAS
1/8th degree spatial
resolution

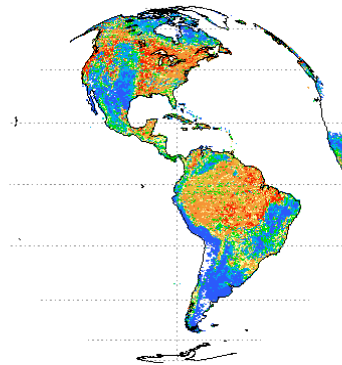
Kumar et al. (2006): Land Information System: An interoperable Framework for High Resolution Land Surface Modeling, Environmental Modeling and Software, Vol 21, pp 1402-1415.

LIS - heritage

- LIS is a land surface modeling and data assimilation system (LDAS)
- Capable of modeling at different spatial scales, globally and regionally



North American LDAS
1/8th degree spatial
resolution

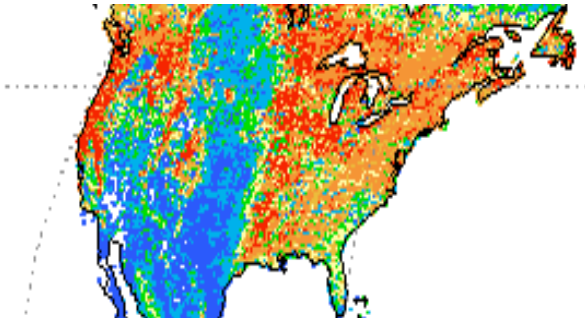


Global LDAS
1/4th degree spatial resolution

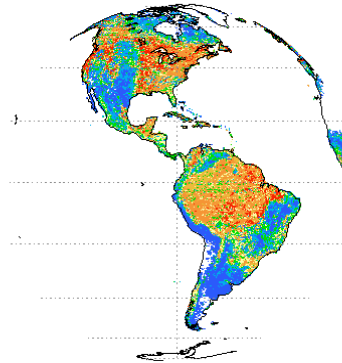
Kumar et al. (2006): Land Information System: An interoperable Framework for High Resolution Land Surface Modeling, Environmental Modeling and Software, Vol 21, pp 1402-1415.

LIS - heritage

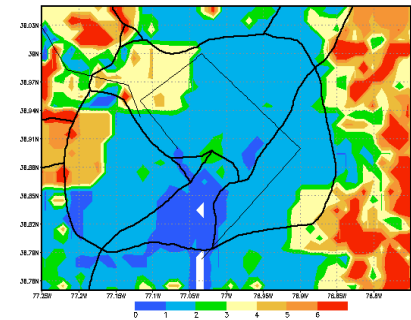
- LIS is a land surface modeling and data assimilation system (LDAS)
- Capable of modeling at different spatial scales, globally and regionally



North American LDAS
1/8th degree spatial
resolution





Global LDAS
1/4th degree spatial resolution



LIS
global, regional, point
up to 1km and finer

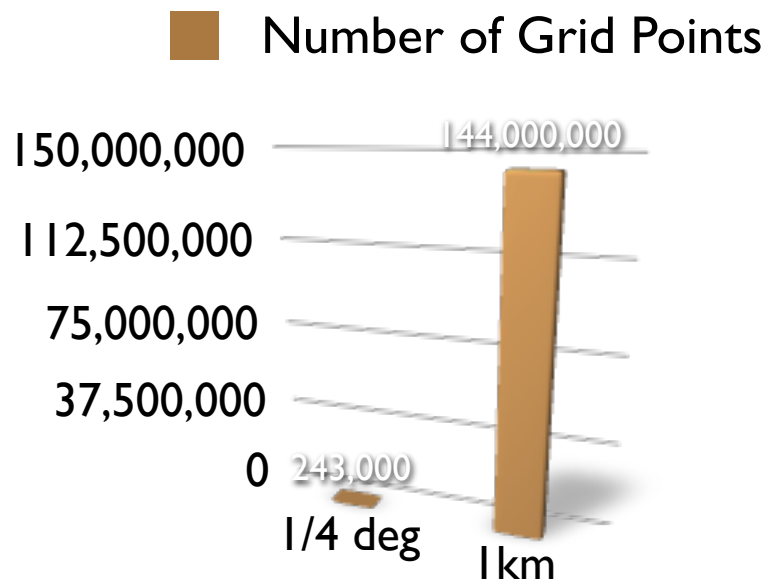
Kumar et al. (2006): Land Information System: An interoperable Framework for High Resolution Land Surface Modeling, Environmental Modeling and Software, Vol 21, pp 1402-1415.

More History



-  Original goal : to enable global land surface modeling at 1km spatial resolution
-  Huge computational challenge

More History

- 📌 Original goal : to enable global land surface modeling at 1km spatial resolution
- 📌 Huge computational challenge

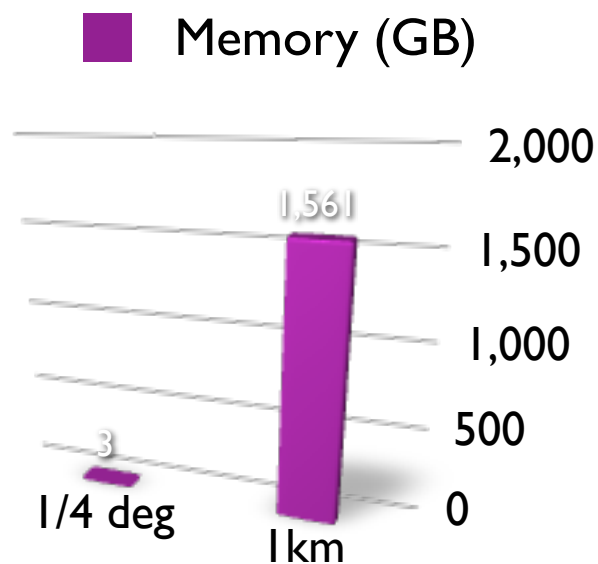


More History



-  Original goal : to enable global land surface modeling at 1km spatial resolution
-  Huge computational challenge

More History

- Original goal : to enable global land surface modeling at 1km spatial resolution
- Huge computational challenge

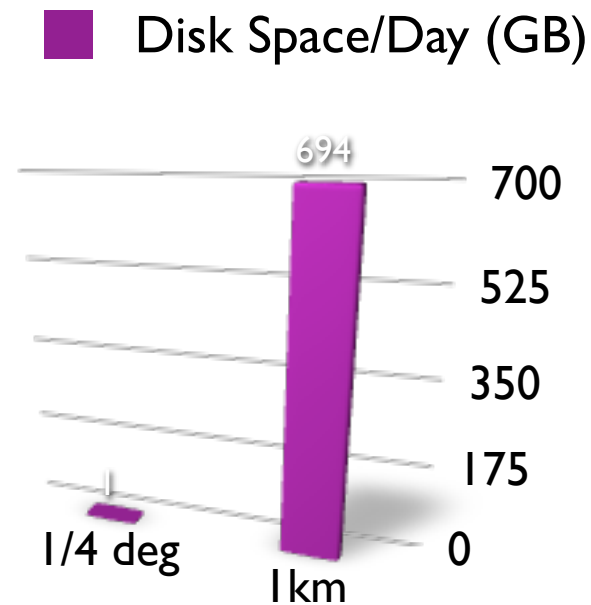


More History



-  Original goal : to enable global land surface modeling at 1km spatial resolution
-  Huge computational challenge

More History

- 📌 Original goal : to enable global land surface modeling at 1km spatial resolution
- 📌 Huge computational challenge

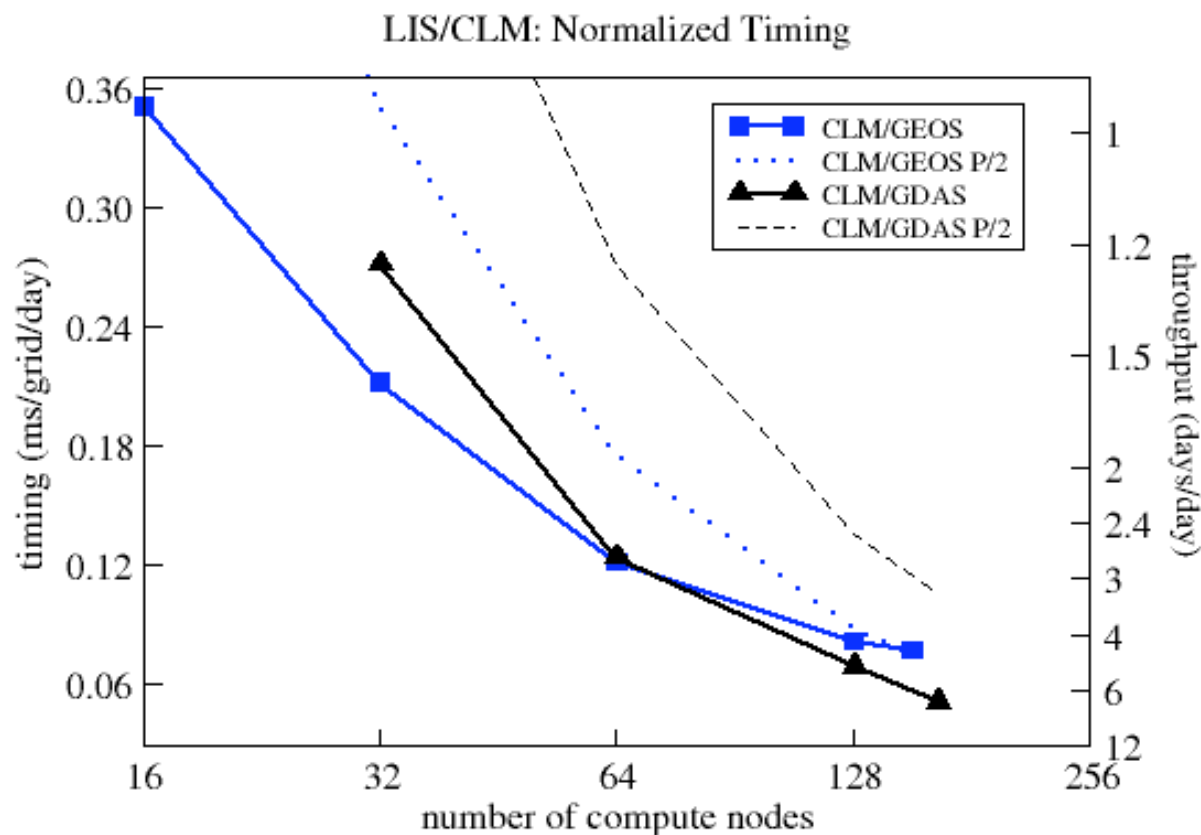


More History

-  Original goal : to enable global land surface modeling at 1km spatial resolution
-  Huge computational challenge

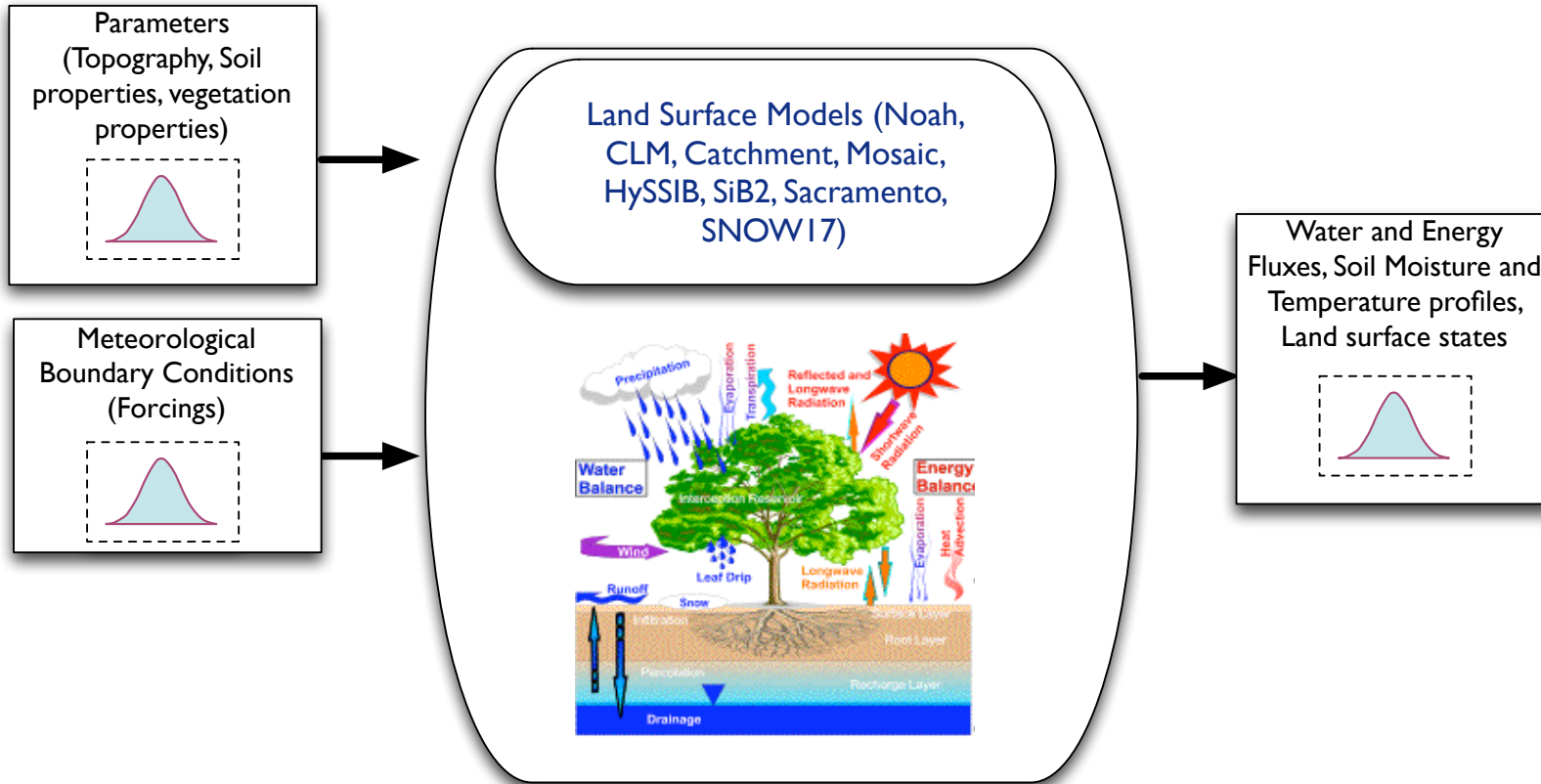
More History

- Original goal : to enable global land surface modeling at 1km spatial resolution
- Huge computational challenge



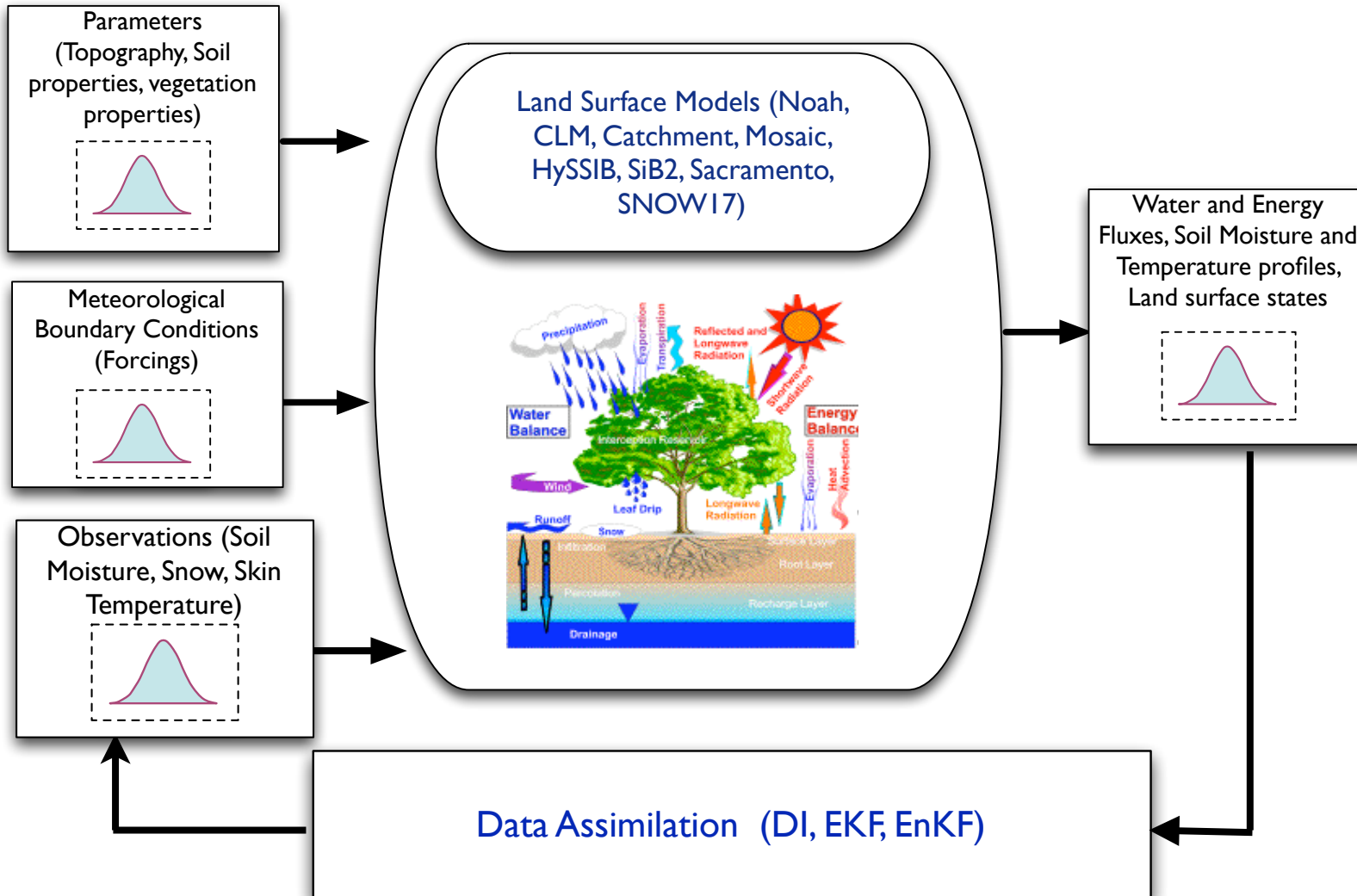
Program Flow

Uncoupled or
Analysis Mode

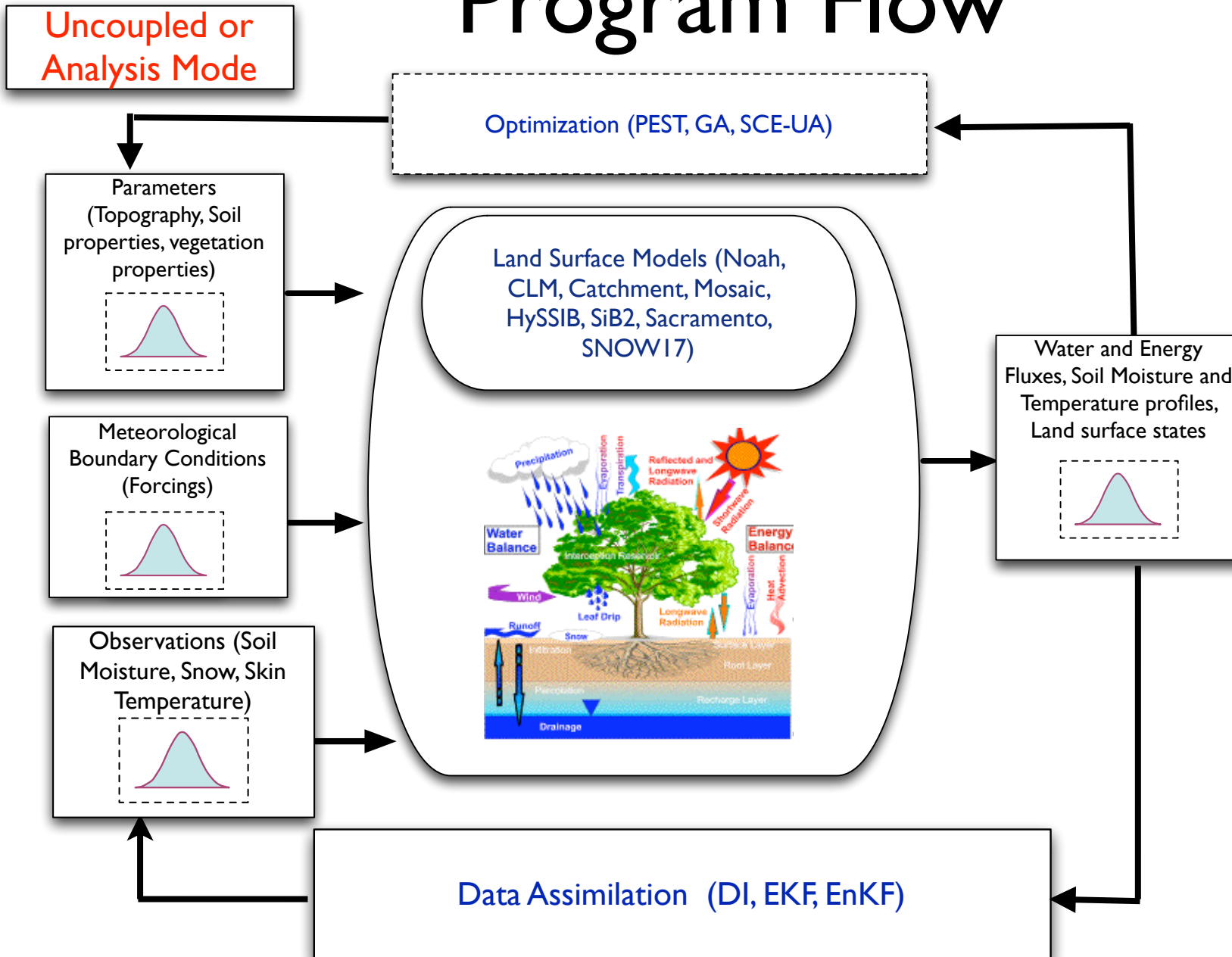


Program Flow

Uncoupled or
Analysis Mode



Program Flow



Program Flow

Uncoupled or
Analysis Mode

Optimization (PEST, GA, SCE-UA)

Coupled or
Forecast Mode

Parameters
(Topography, Soil
properties, vegetation
properties)



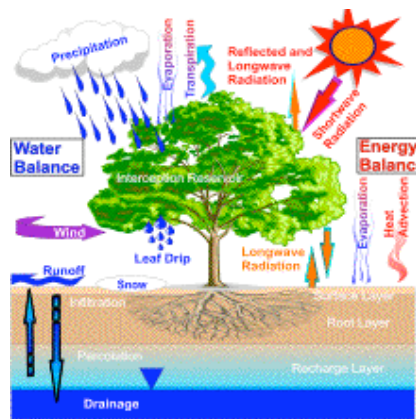
Meteorological
Boundary Conditions
(Forcings)



Observations (Soil
Moisture, Snow, Skin
Temperature)



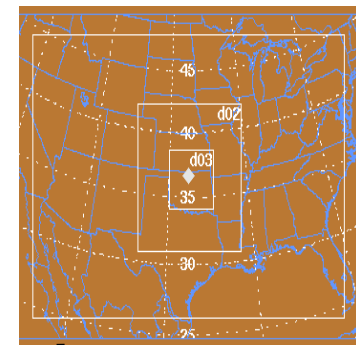
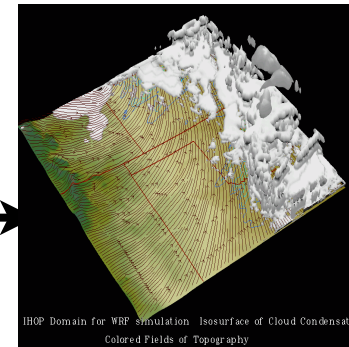
Land Surface Models (Noah,
CLM, Catchment, Mosaic,
HySSIB, SiB2, Sacramento,
SNOW17)



Water and Energy
Fluxes, Soil Moisture and
Temperature profiles,
Land surface states



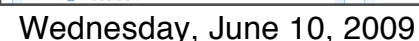
LIS - WRF
Interface



WRF

Data Assimilation (DI, EKF, EnKF)

<http://modelingguru.nasa.gov>



Software Requirements

- ☒ Fortran 90/95 compiler (g95 will not work for LIS5.0)
 - ☐ preferred : intel, pgi, lahey, absoft
- ☒ C compiler
- ☐ MPI - if parallel processing capability is desired
- ☒ Earth System Modeling Framework (ESMF)
 - ☐ 2.2.2rp3 - for LIS5.0
 - ☐ 3.1.0r - for LIS 6.0
- ☐ LIS supports Grib I, NETCDF, HDF formats



LIS Documentation

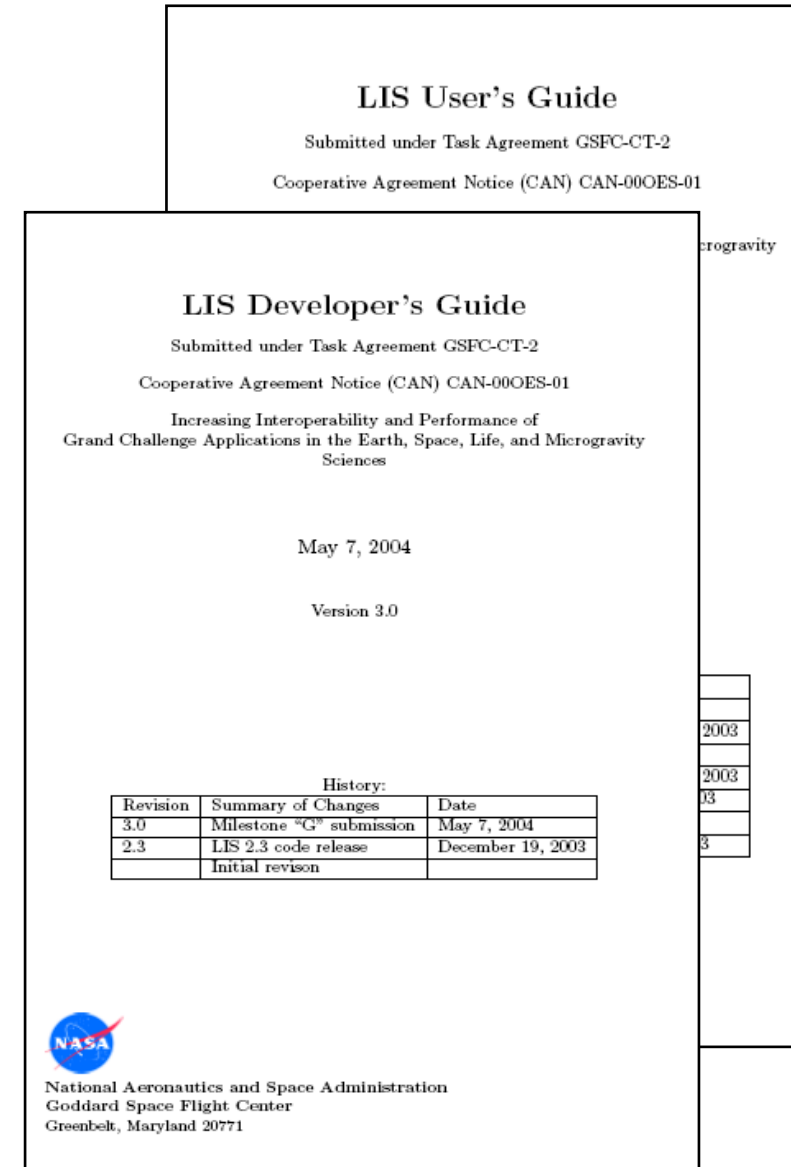
User's guide

Step-by-step instructions on how to build the LIS code

Developer's guide

Instructions on how to bring in new functionalities (LSMs, forcing schemes, Data Assimilation, parameter data, etc.)

Reference manual



Getting LIS source

- 📌 Use the subversion repository (<https://flood.gsfc.nasa.gov>)
- 📌 Apply for an account (James.V.Geiger@nasa.gov)
- 📌 Request a “Project Release” of the LIS code (<http://lis.gsfc.nasa.gov/register.shtml>)
- 📌 Check out the code



APPROVED

LIS source code repository



Helpful links

<http://subversion.tigris.org/>

<http://svnbook.red-bean.com/>



Check out the LIS code:

svn co <https://flood.gsfc.nasa.gov/svn/5/public/> src

LIS source code repository



Helpful links

<http://subversion.tigris.org/>

<http://svnbook.red-bean.com/>



Check out the LIS code:

svn co <https://flood.gsfc.nasa.gov/svn/5/public/> src

LIS source code repository



Helpful links

<http://subversion.tigris.org/>

<http://svnbook.red-bean.com/>



Check out the LIS code:

```
svn co https://flood.gsfc.nasa.gov/svn/5/public/ src
```



LIS source code repository



Helpful links

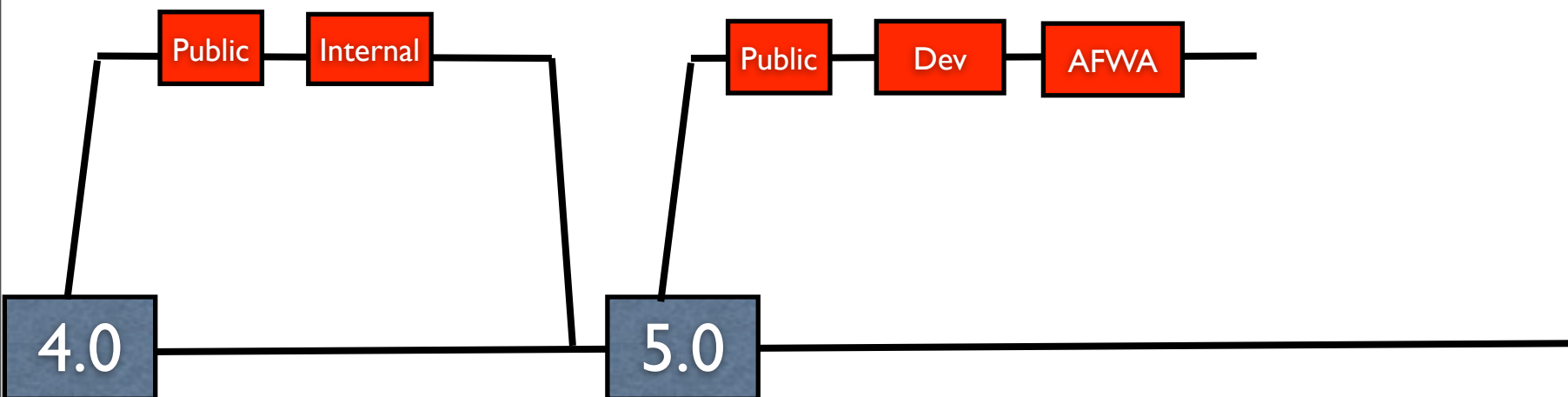
<http://subversion.tigris.org/>

<http://svnbook.red-bean.com/>



Check out the LIS code:

svn co <https://flood.gsfc.nasa.gov/svn/5/public/> src



LIS source code repository



Helpful links

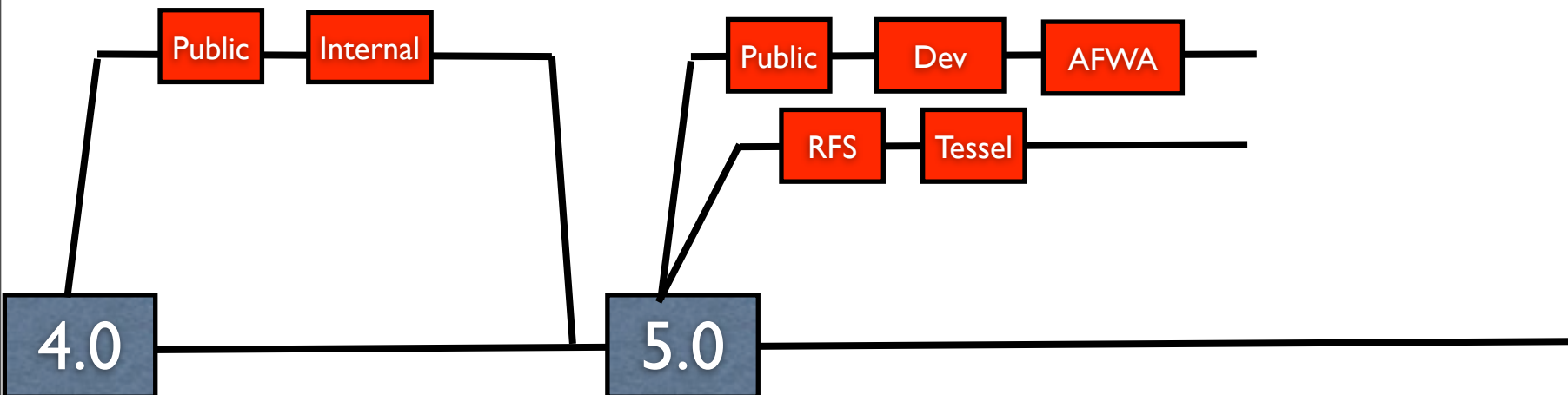
<http://subversion.tigris.org/>

<http://svnbook.red-bean.com/>



Check out the LIS code:

svn co <https://flood.gsfc.nasa.gov/svn/5/public/> src



LIS source code repository



Helpful links

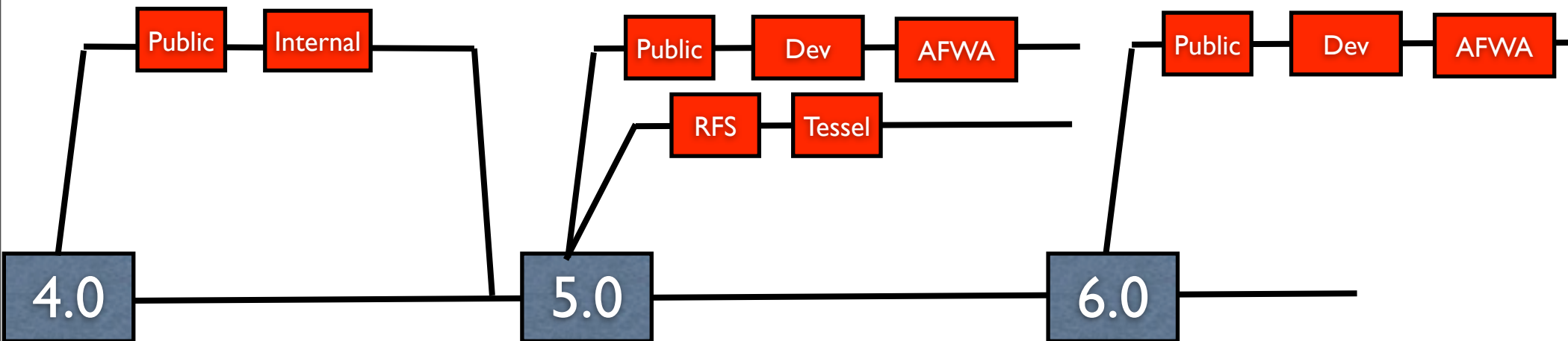
<http://subversion.tigris.org/>

<http://svnbook.red-bean.com/>



Check out the LIS code:

svn co <https://flood.gsfc.nasa.gov/svn/5/public/> src



LIS source code repository



Helpful links

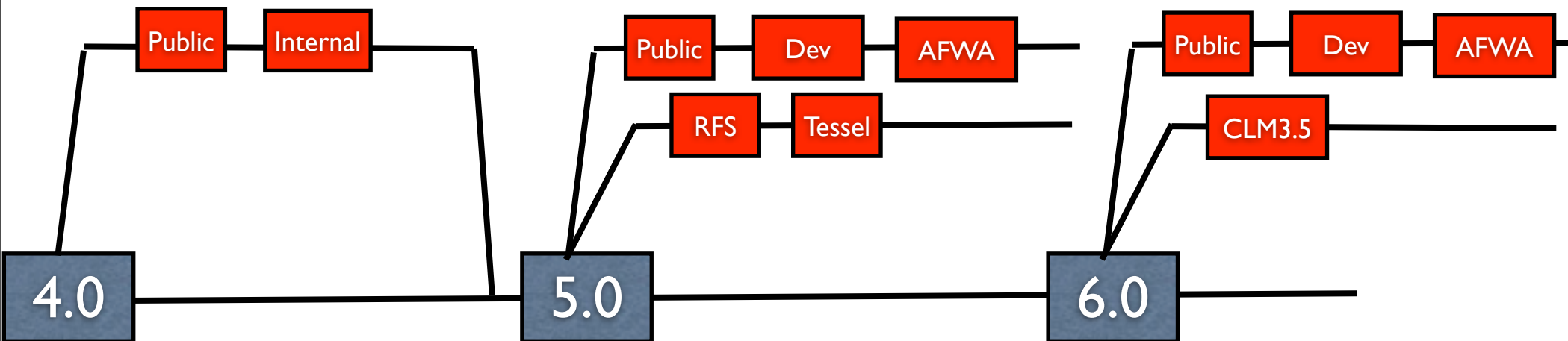
<http://subversion.tigris.org/>

<http://svnbook.red-bean.com/>

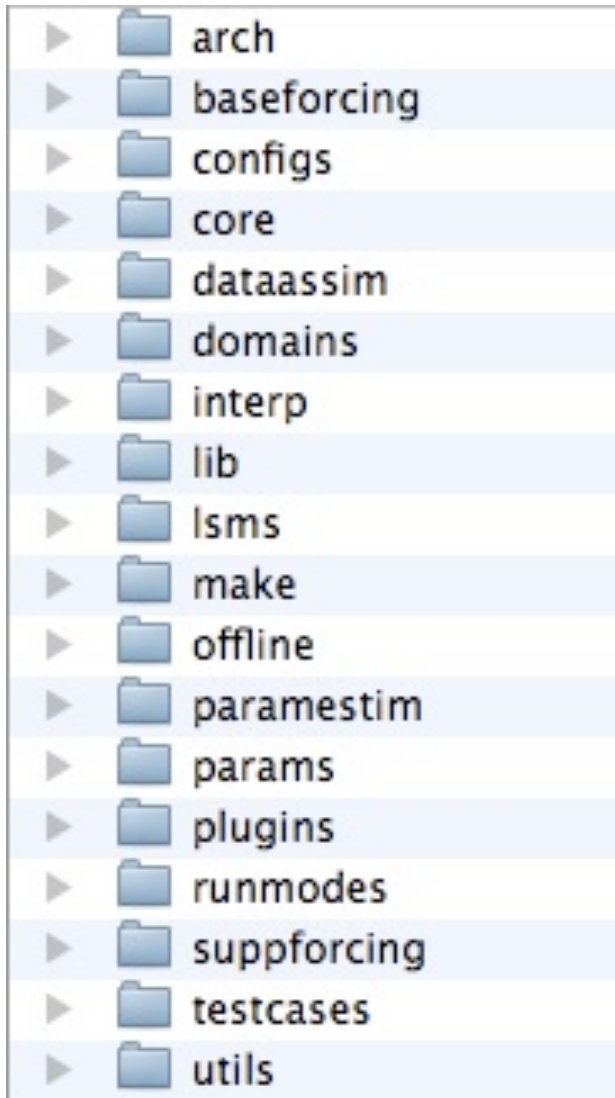


Check out the LIS code:

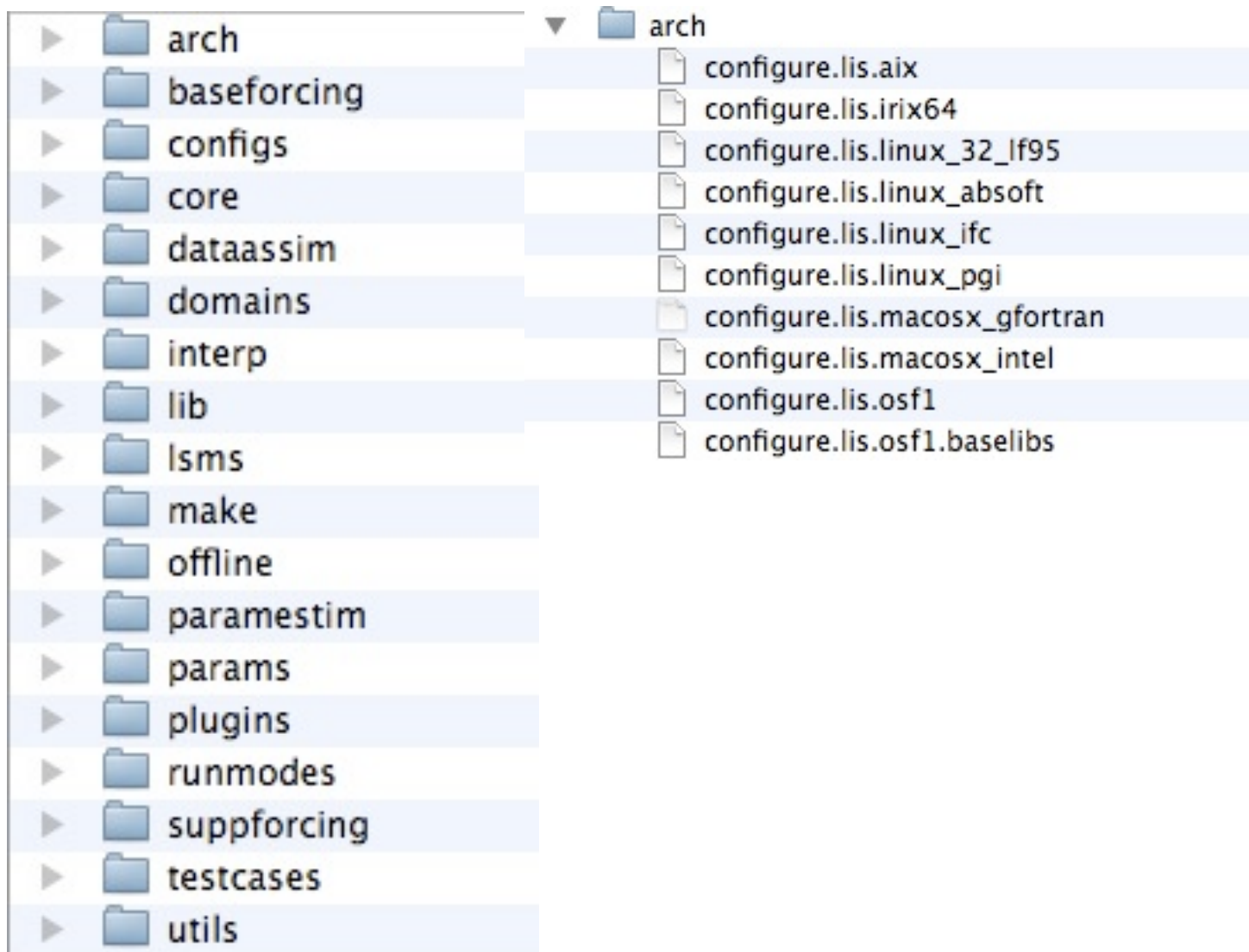
svn co <https://flood.gsfc.nasa.gov/svn/5/public/> src



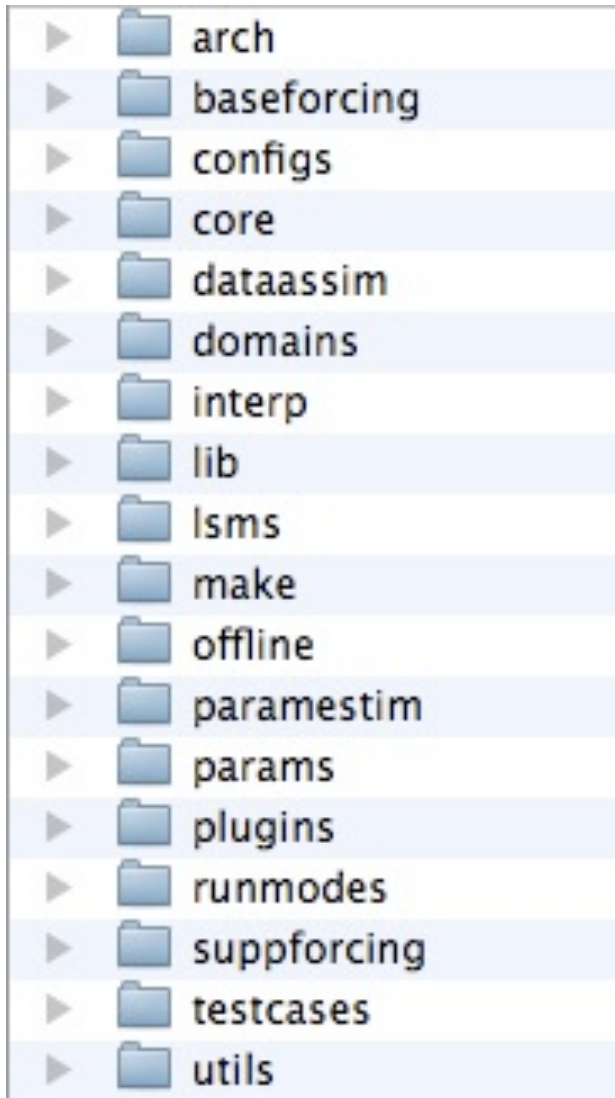
Code organization



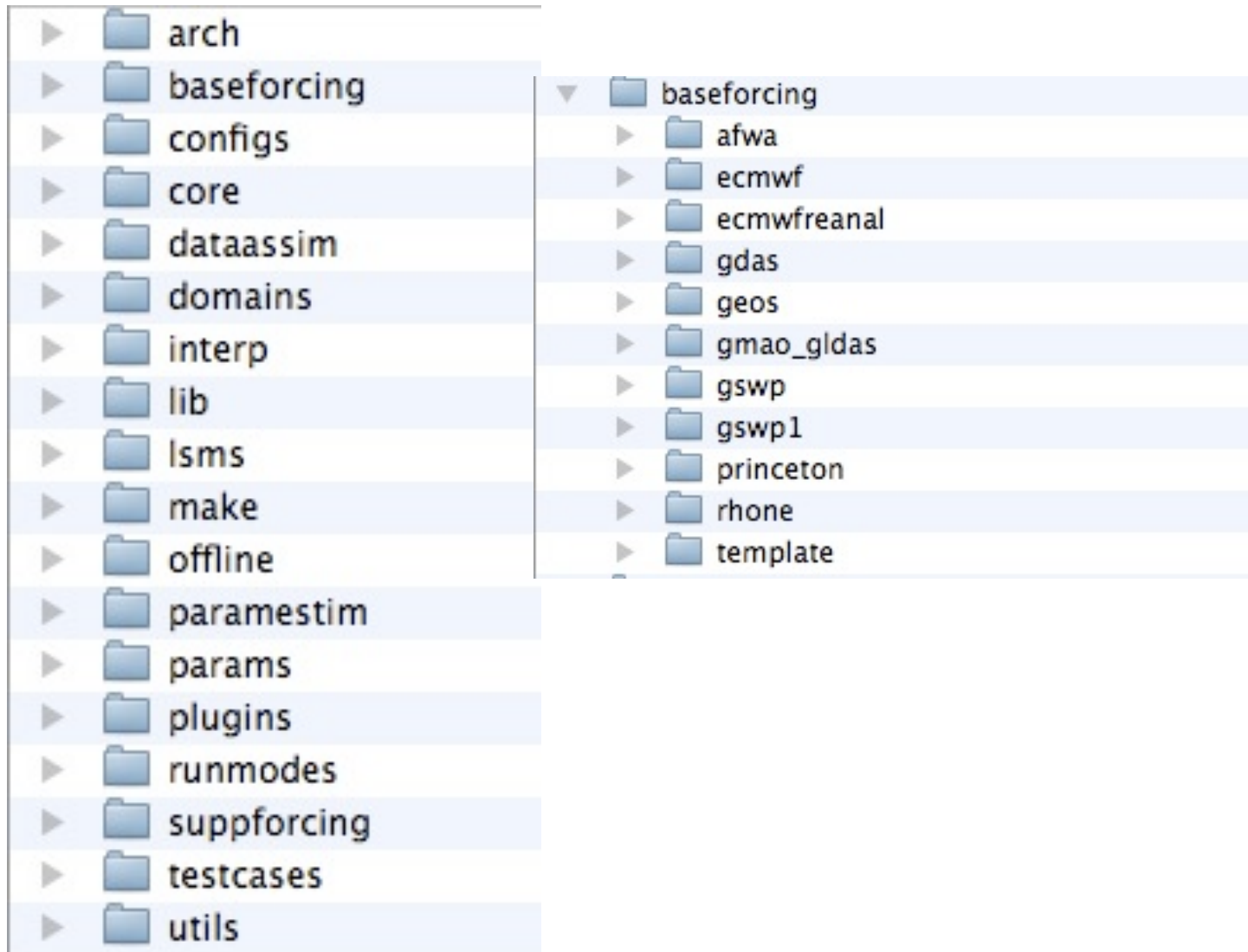
Code organization



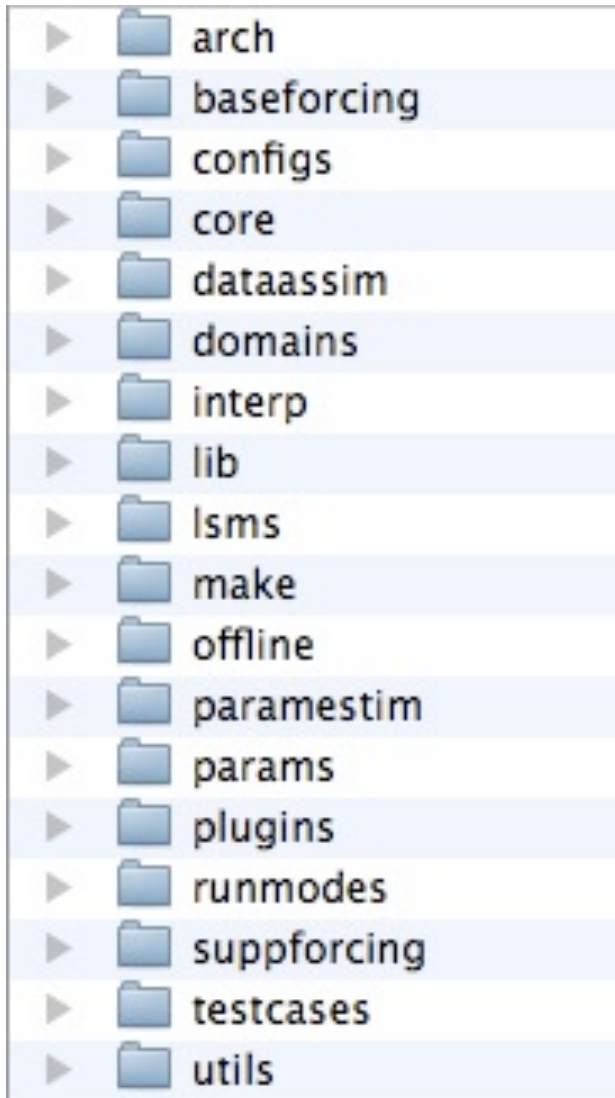
Code organization



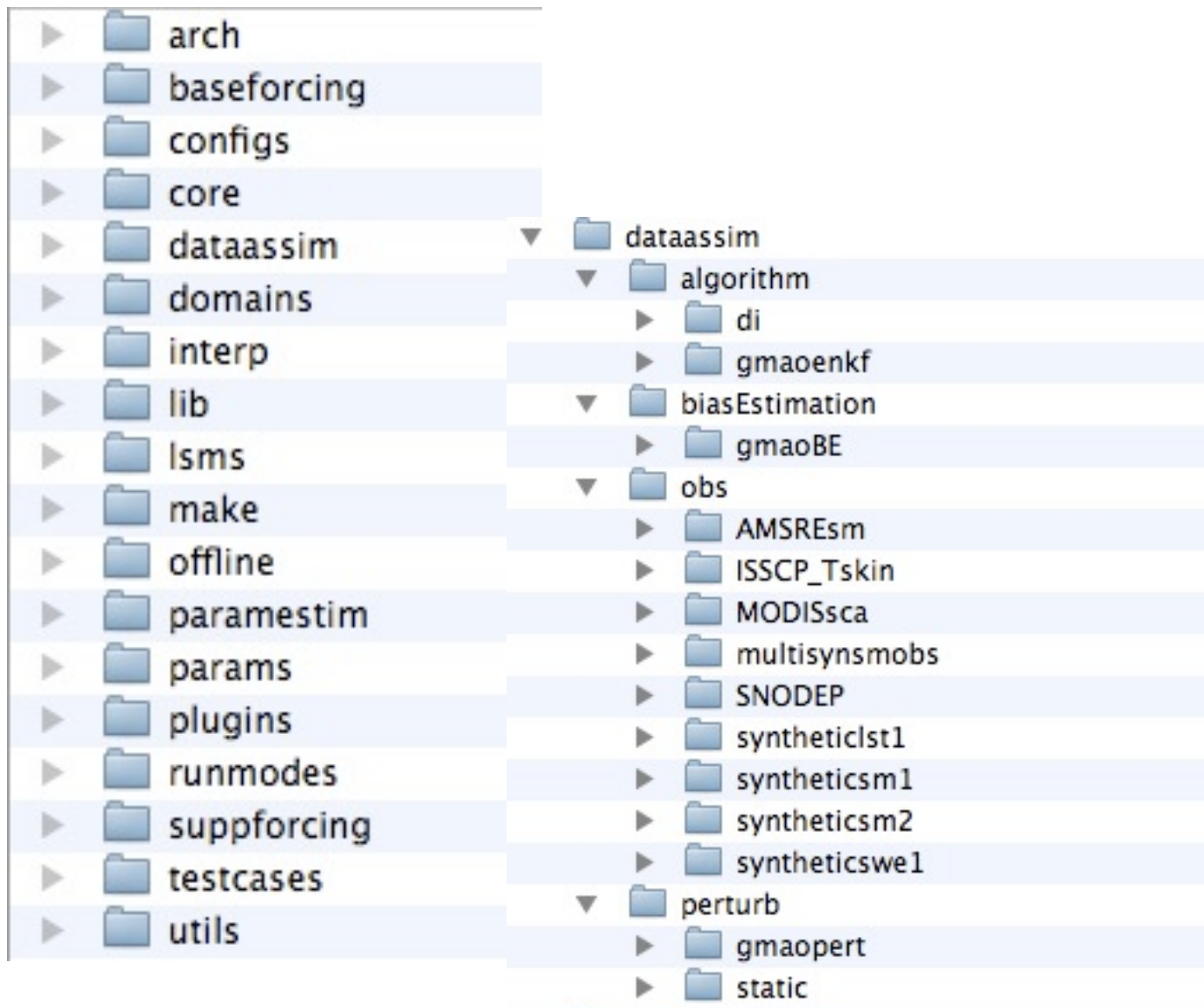
Code organization



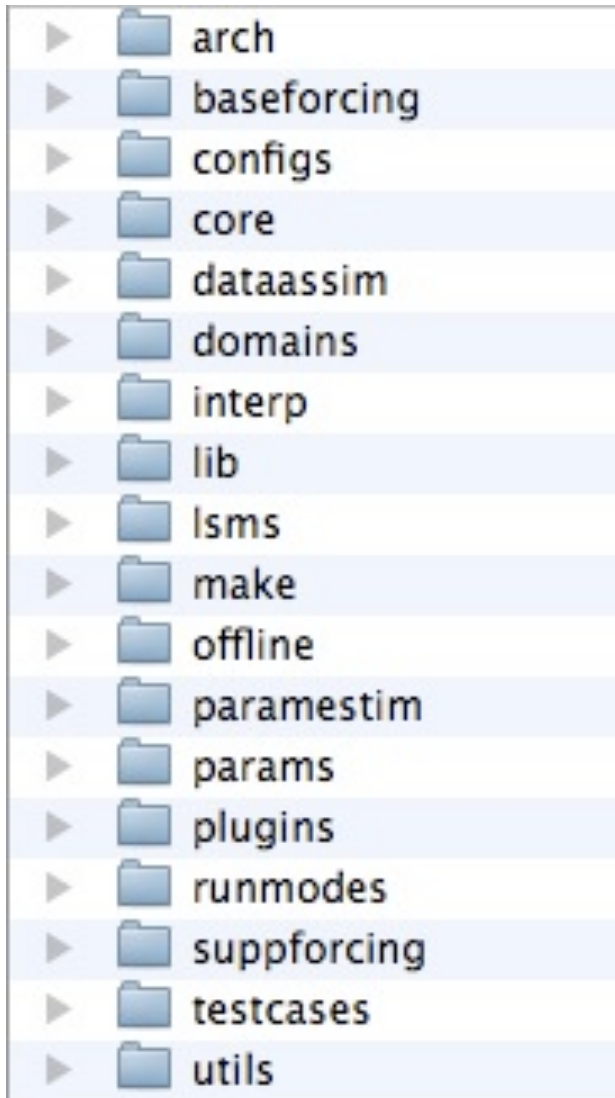
Code organization



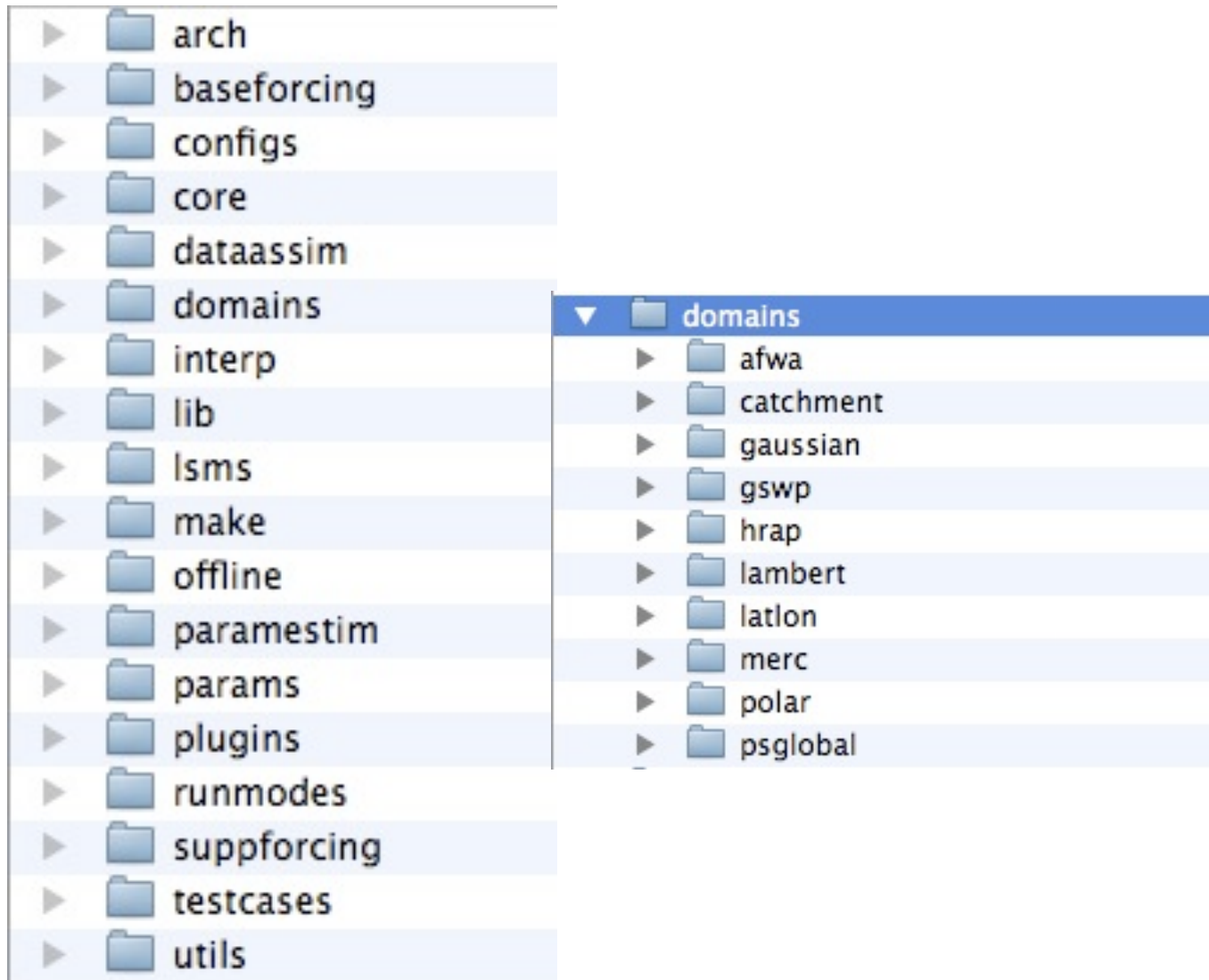
Code organization



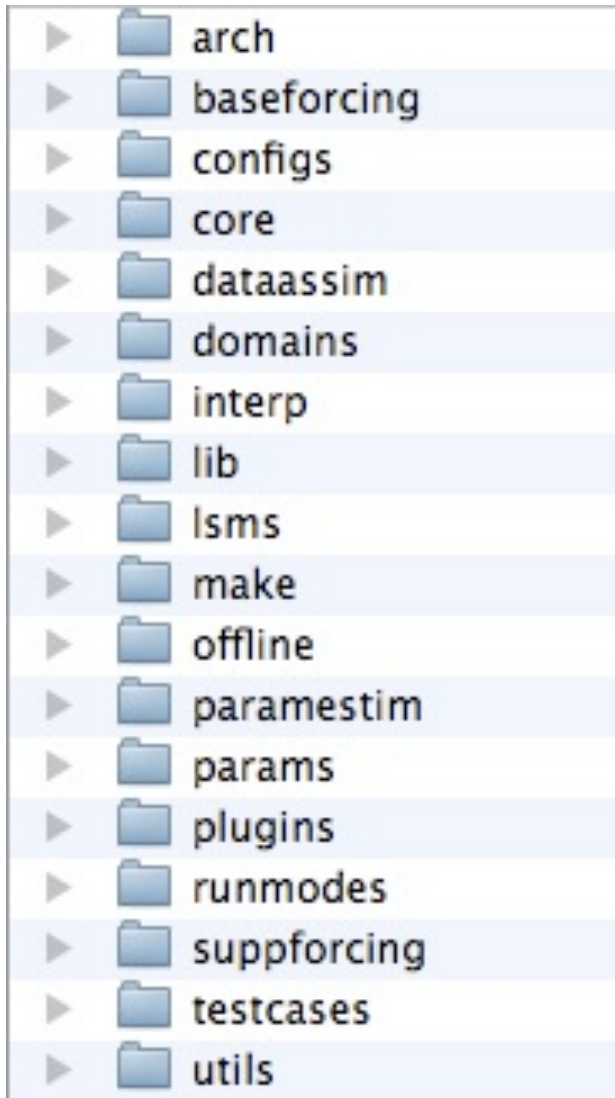
Code organization



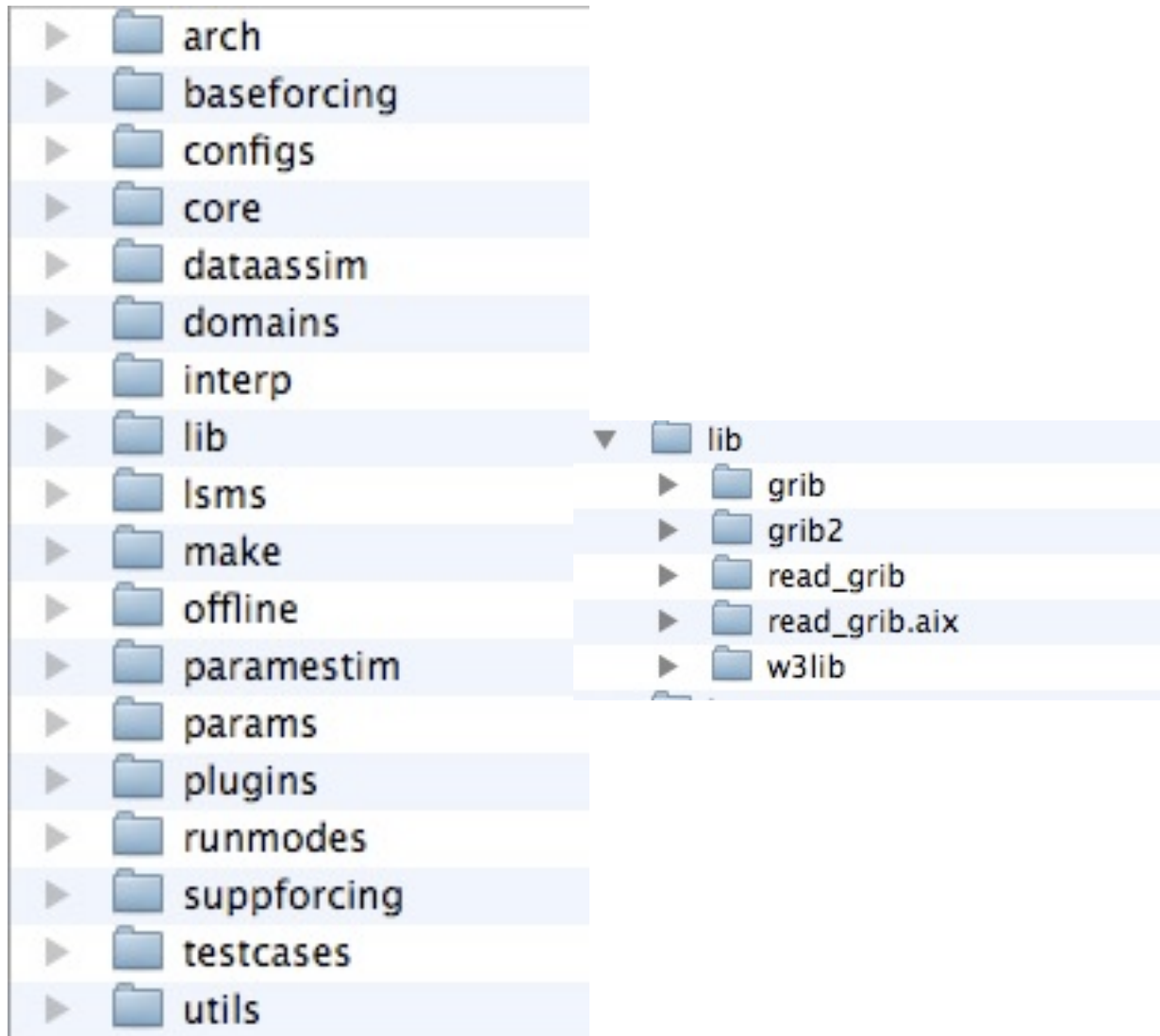
Code organization



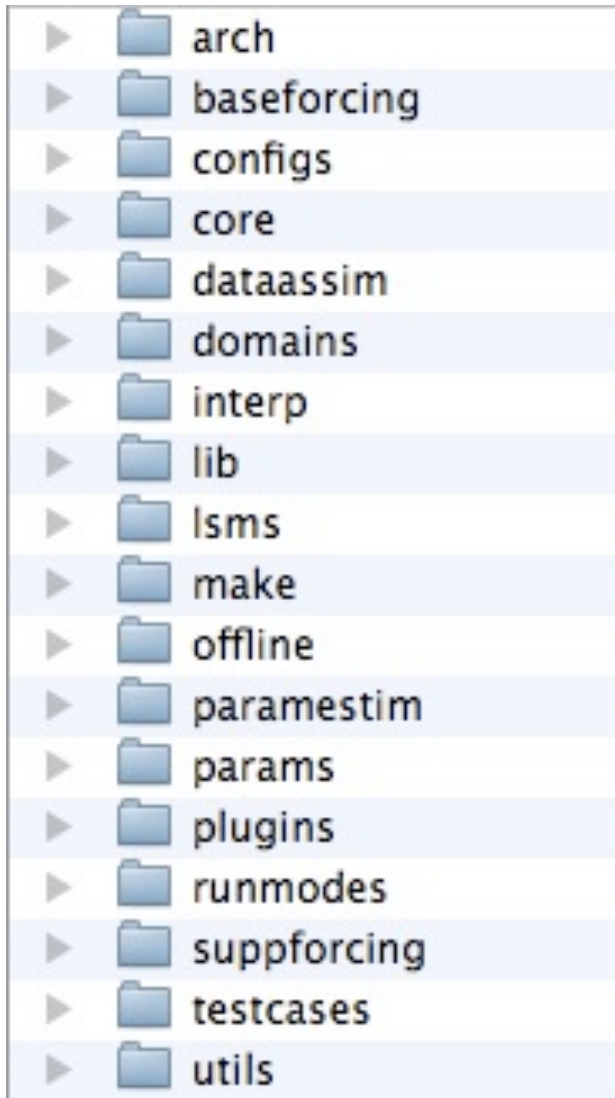
Code organization



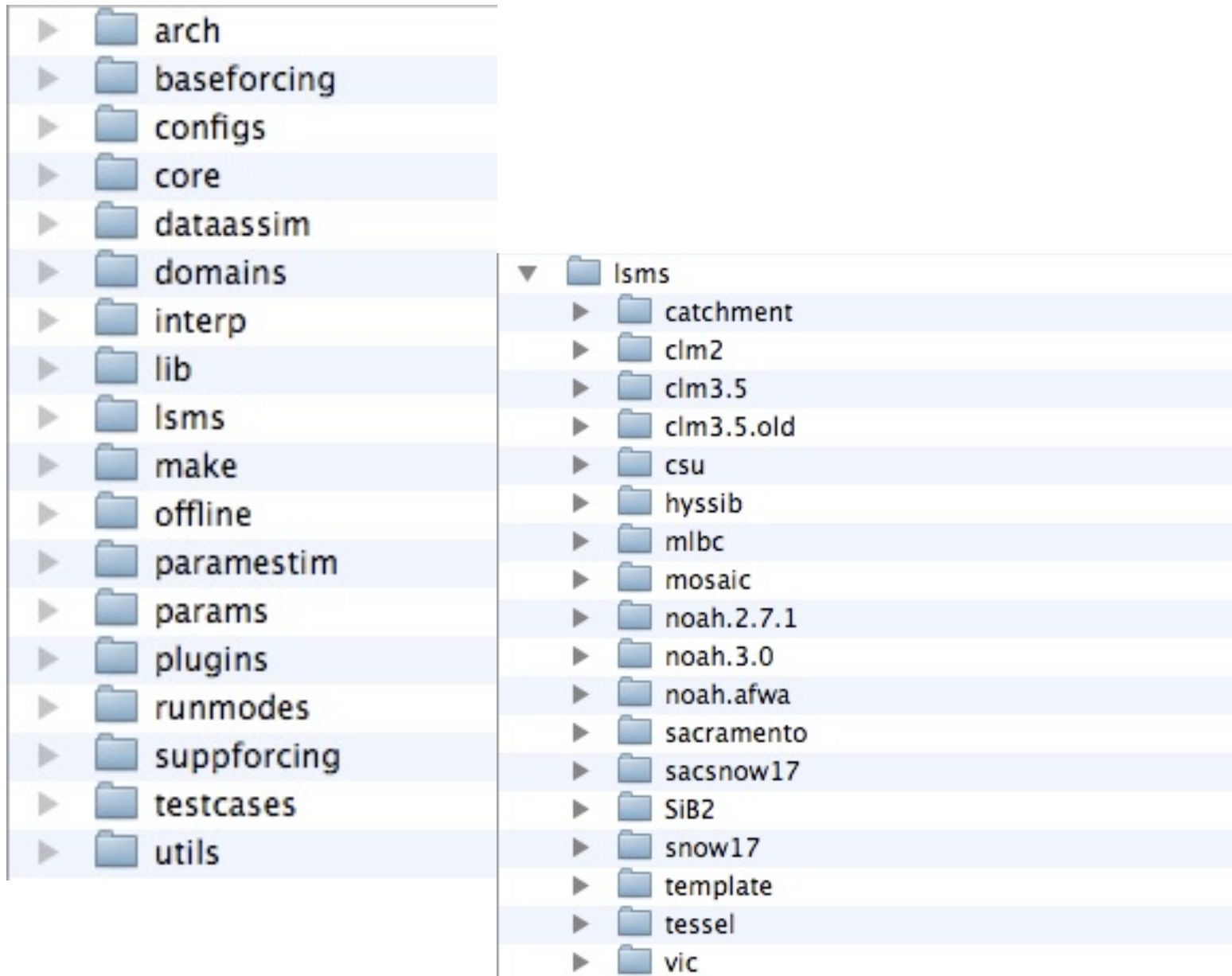
Code organization



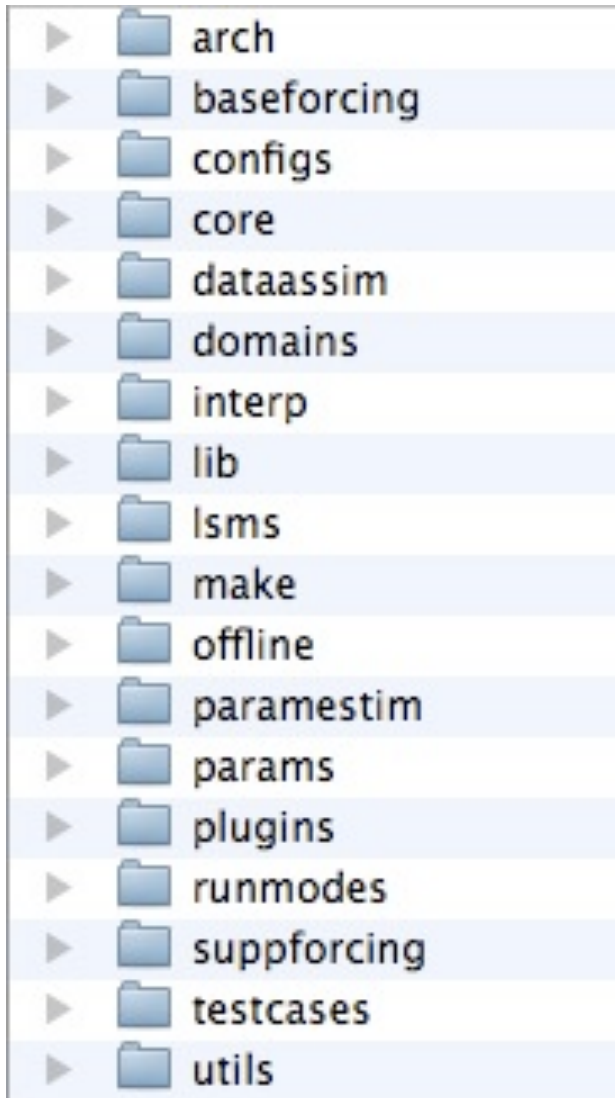
Code organization



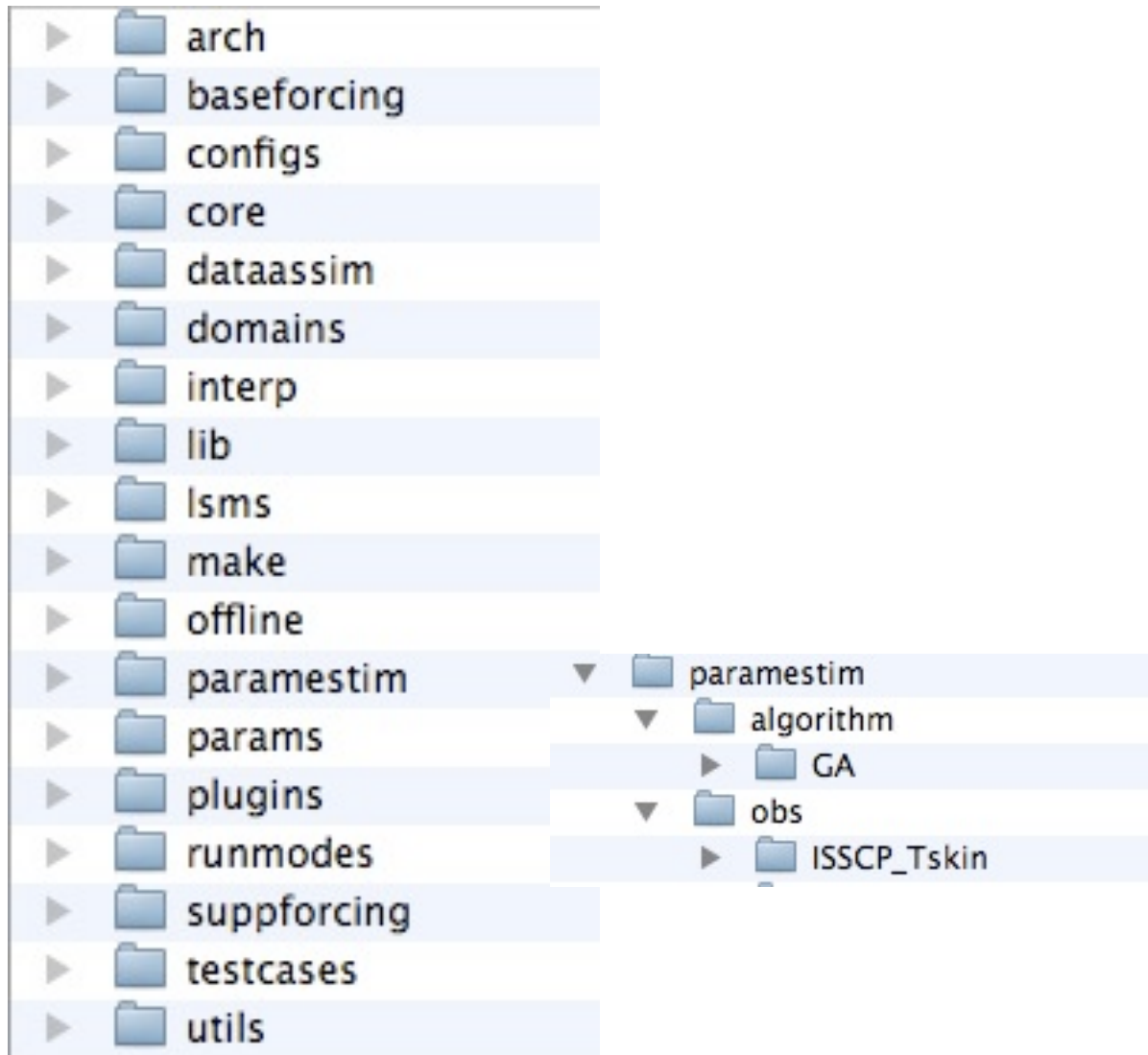
Code organization



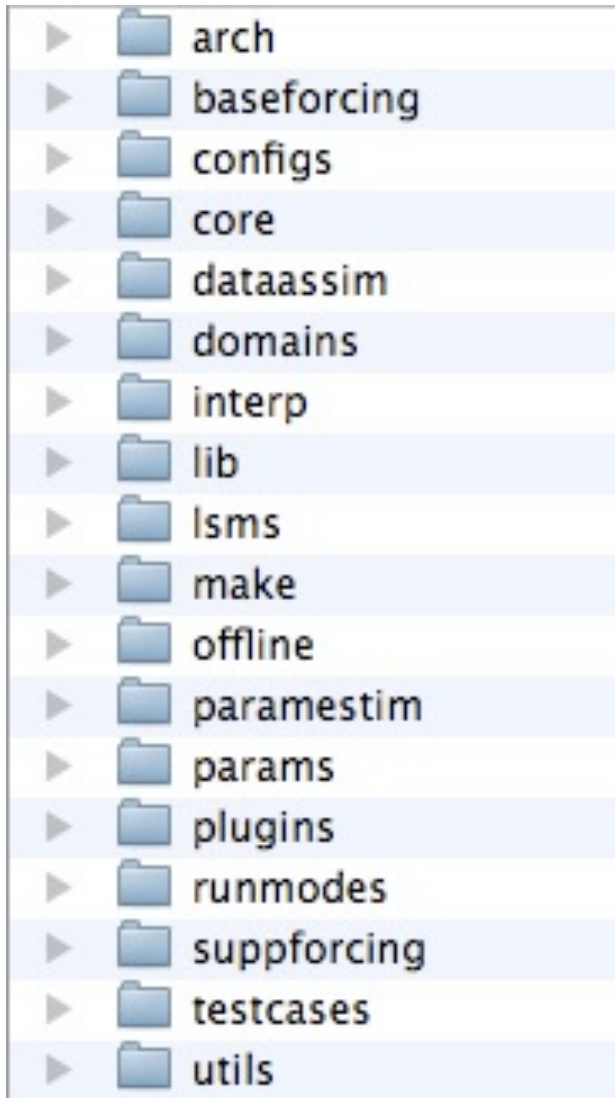
Code organization



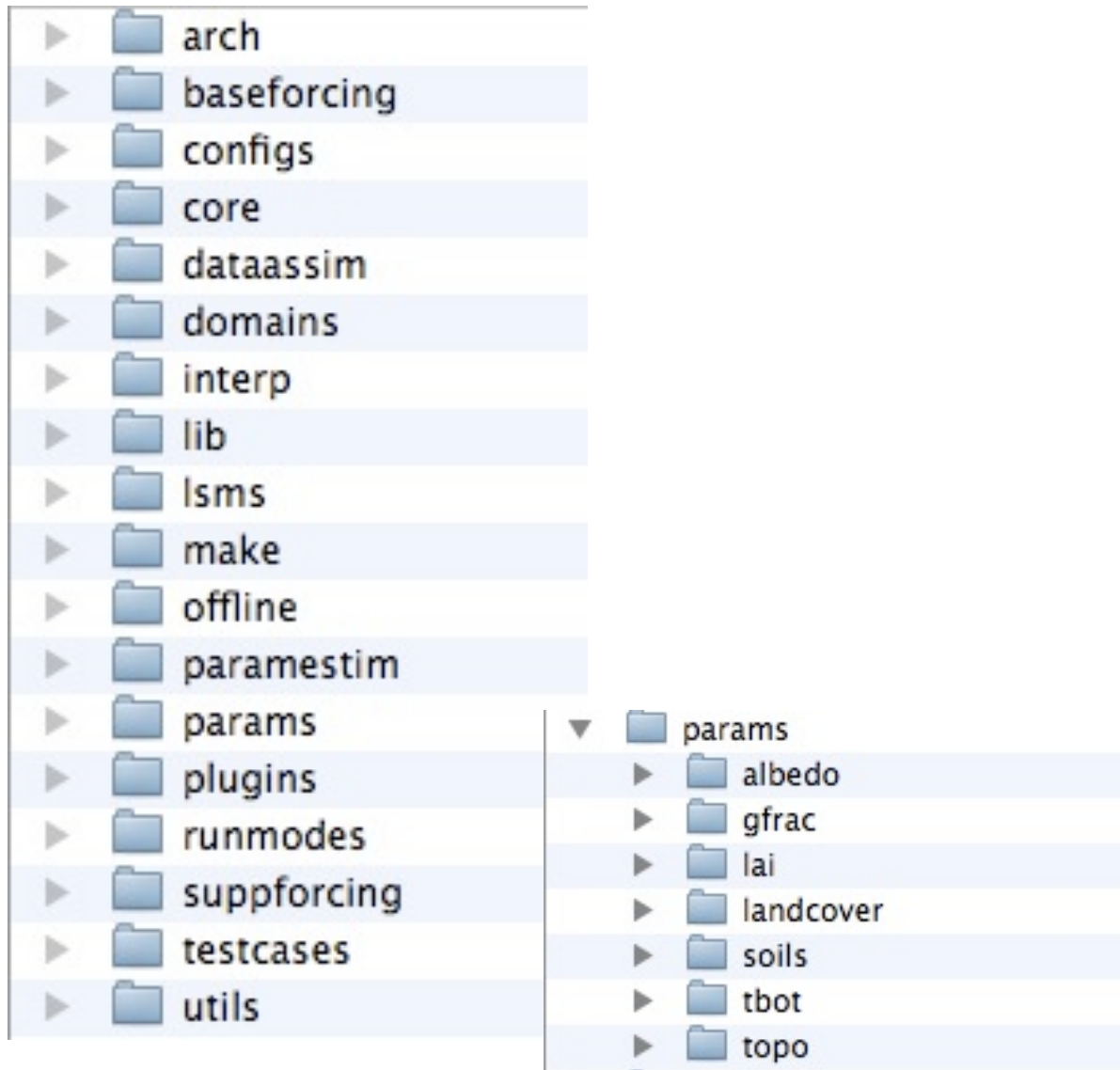
Code organization



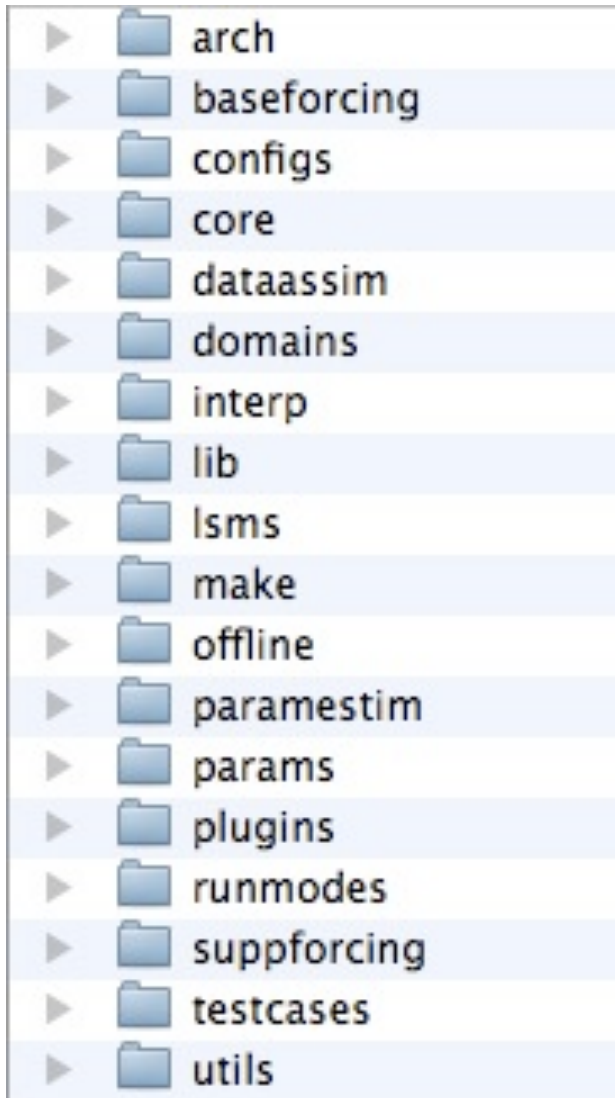
Code organization



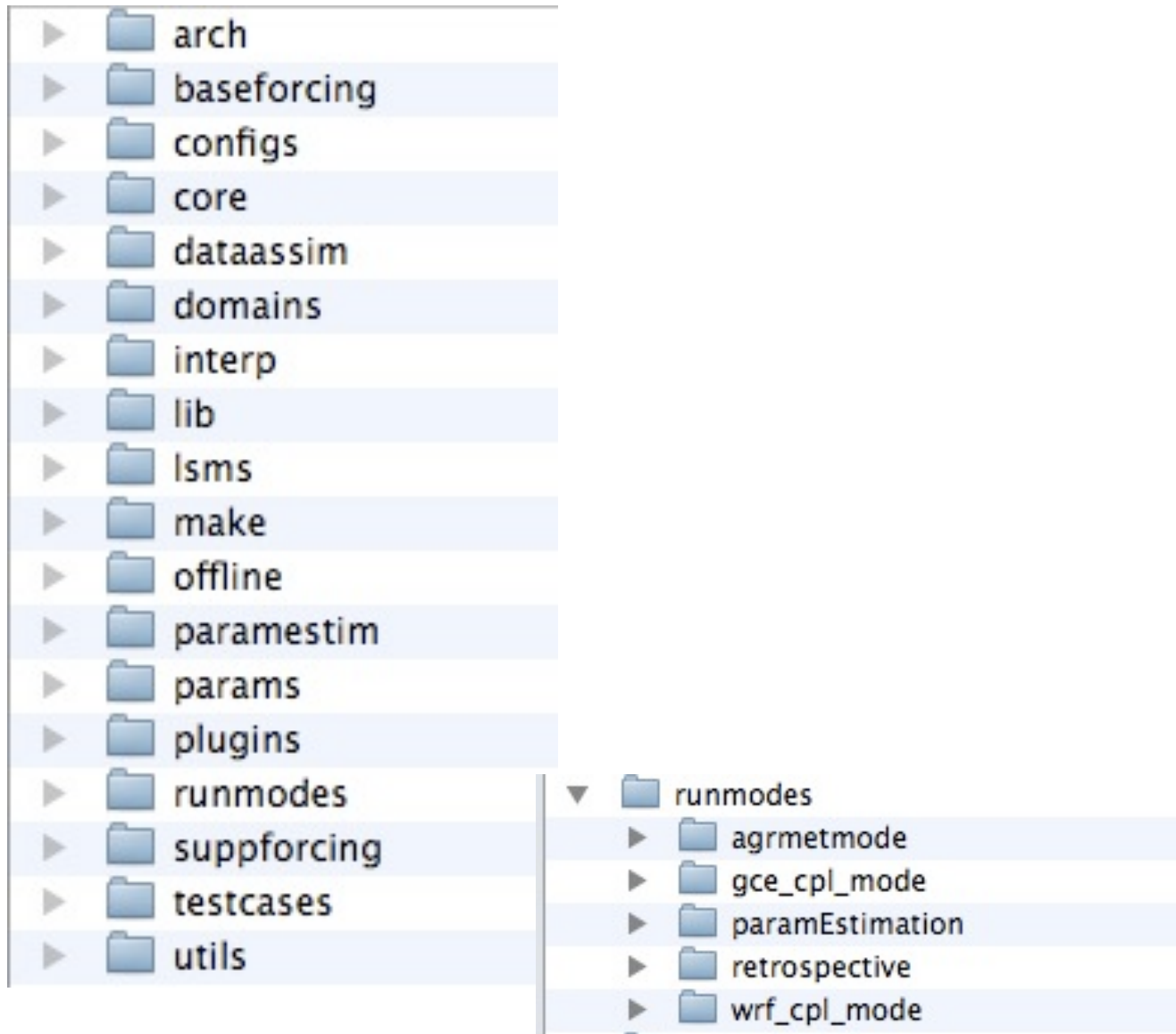
Code organization



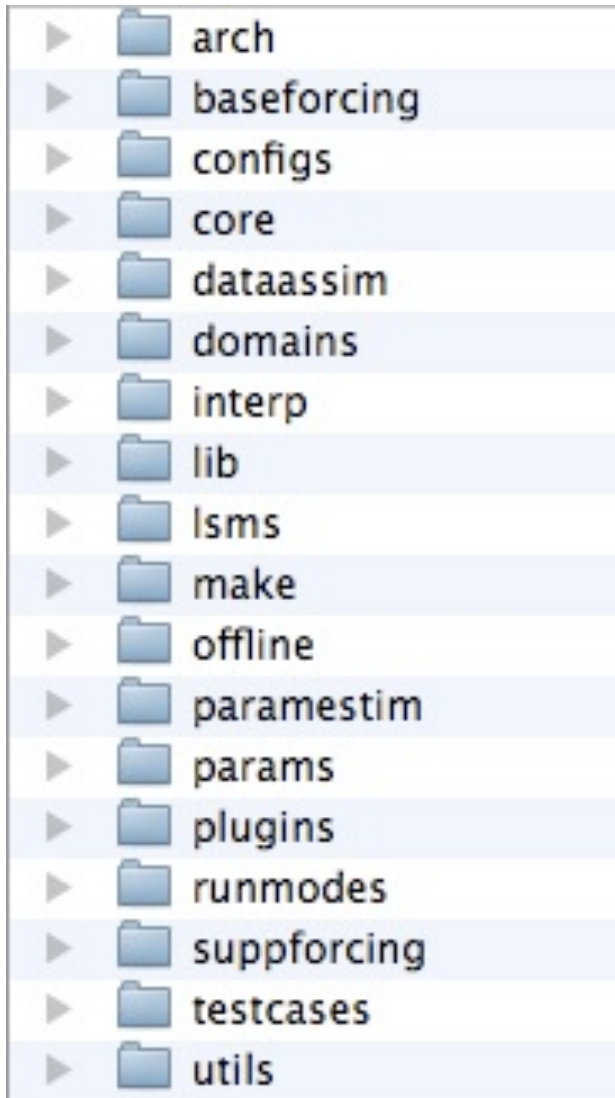
Code organization



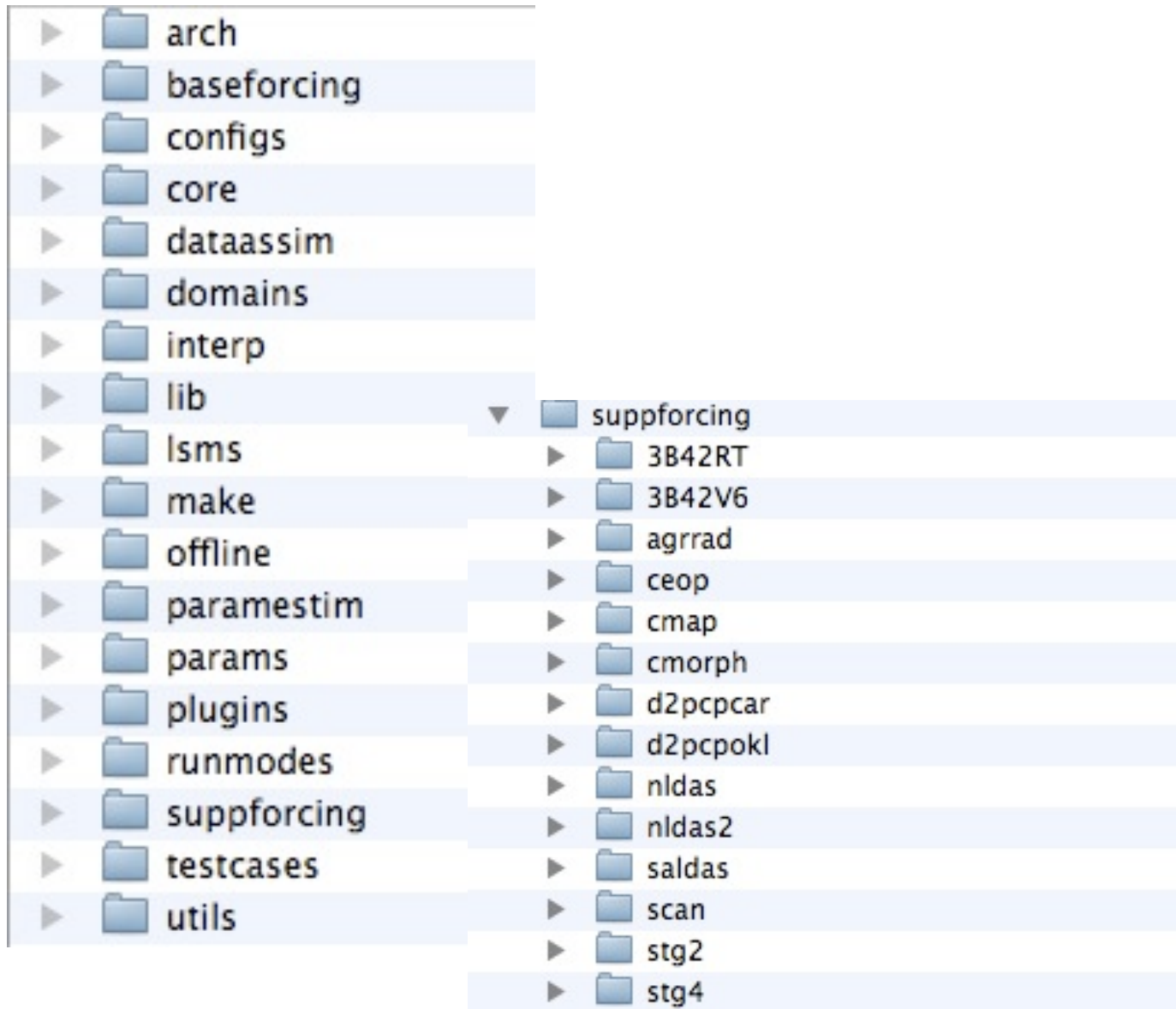
Code organization



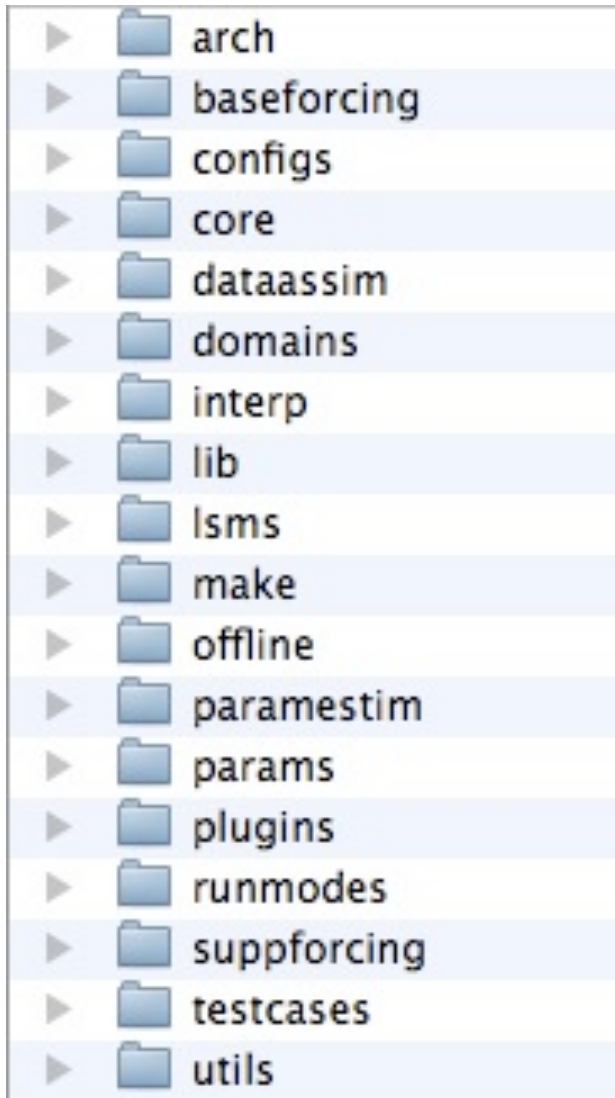
Code organization



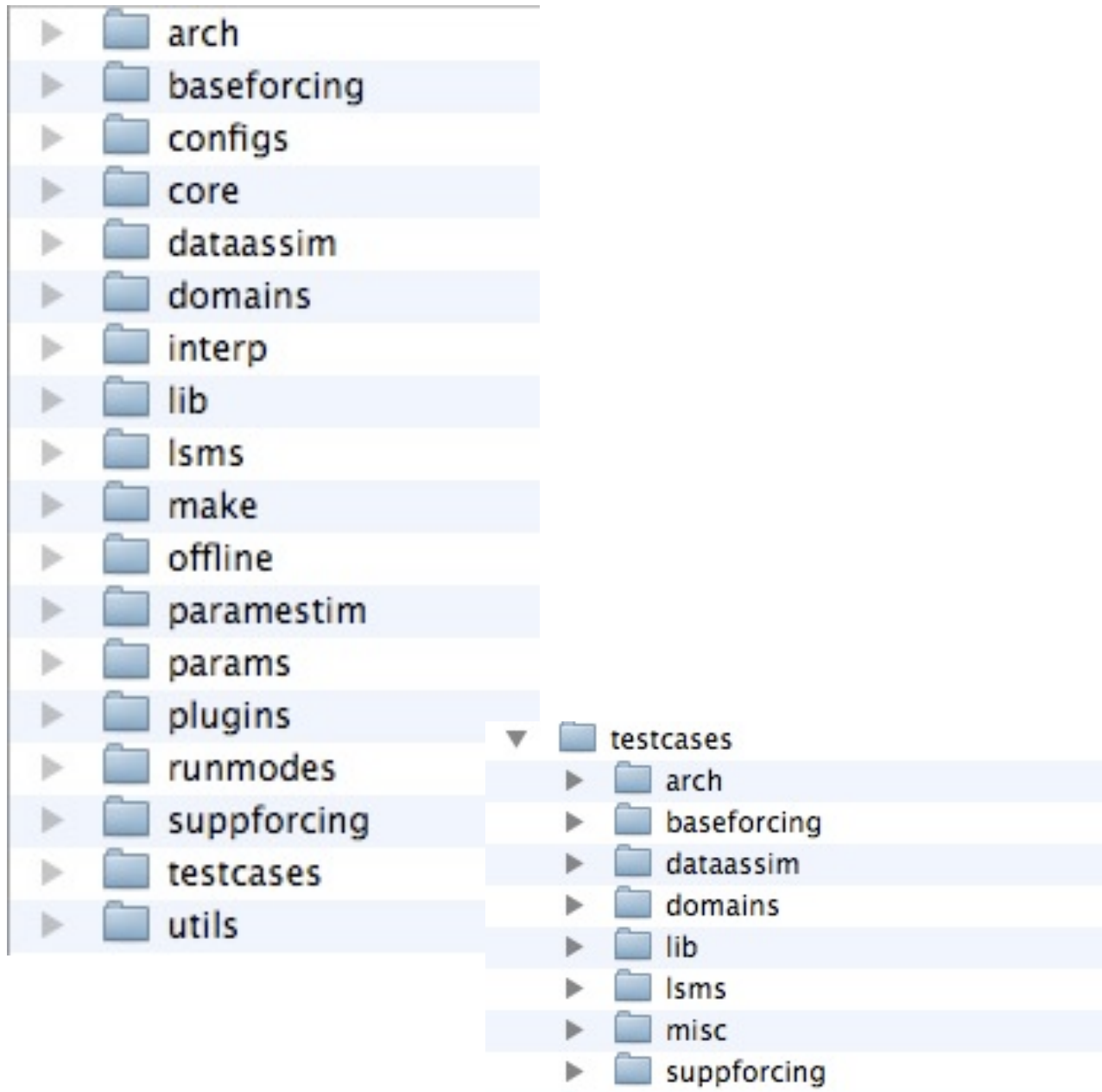
Code organization



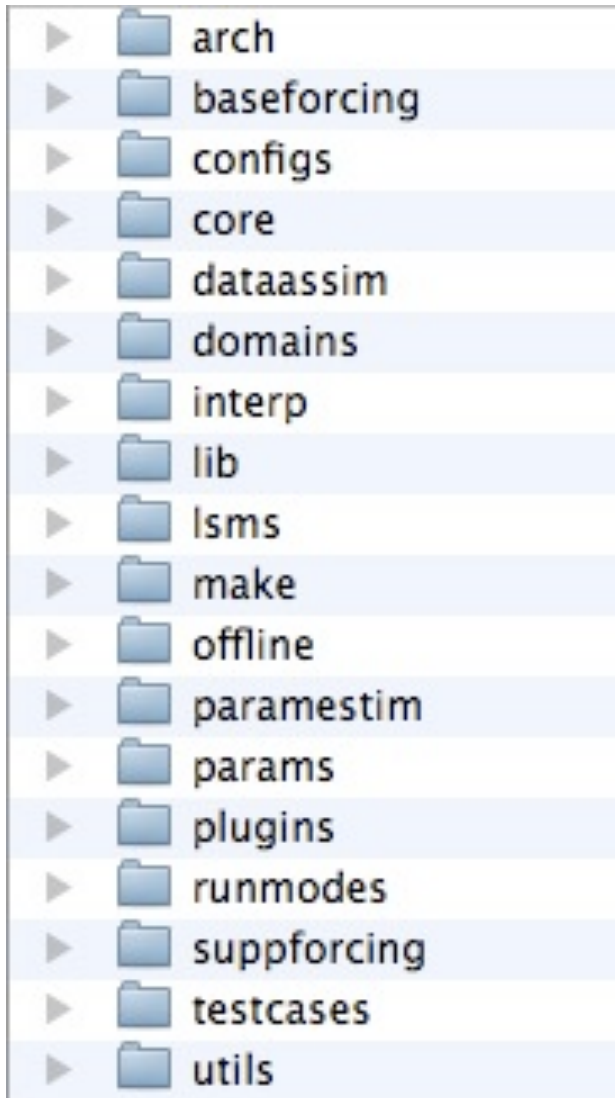
Code organization



Code organization



Code organization



Compiling LIS code



1. Build the libraries

set the environment variable LIS_ARCH

Grib1 (w3lib, read_grib, grib) - provided with the source code (under src/lib)

ESMF - download the required version from <http://www.esmf.ucar.edu>

2. Build the dependency generator

src/make/MAKDEP

3. Specify the configure.lis file that contains the architecture/compiler specific flags

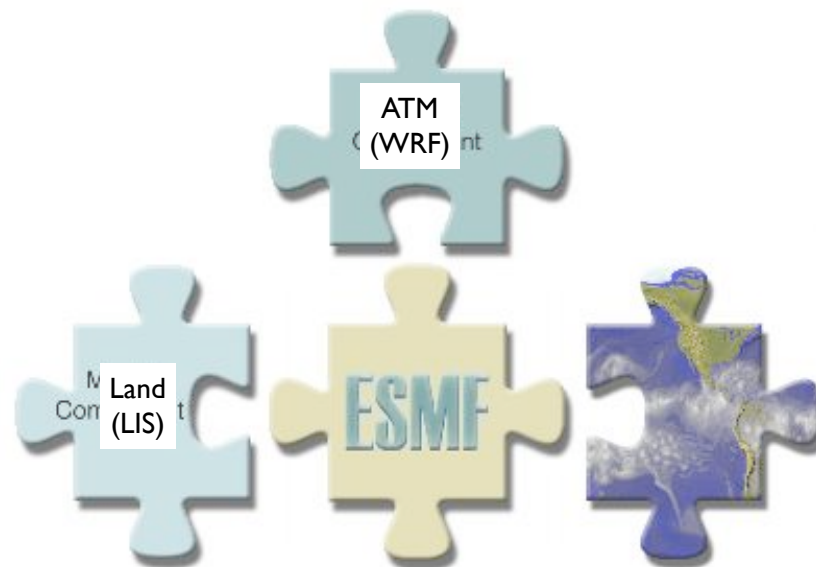
Sample files under src/arch

4. Edit the misc.h file

to turn on/off parallel processing, NETCDF, HDF

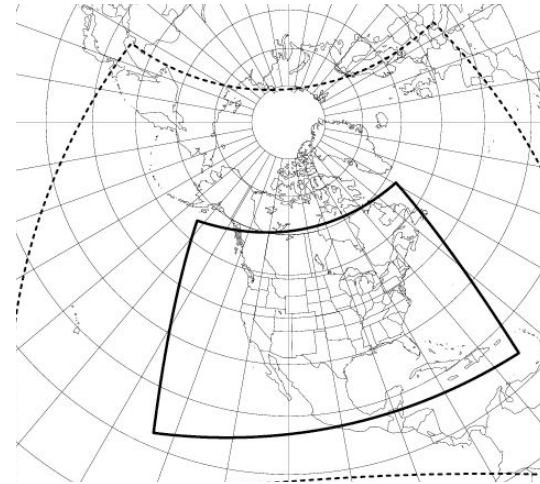
What is ESMF?

- Software for building and coupling weather, climate and related models
- Provides representations of Earth system grids, tools for mapping between them in multiprocessor environment
- Includes toolkits for building applications: time manager, error handling, resource management, parallel communications
- An application once it is wrapped with ESMF is known as a “Gridded component”
- Gridded components are coupled using “coupler components”



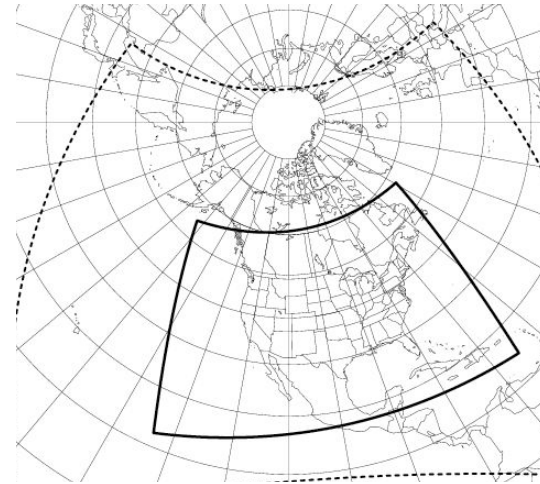
Key ESMF objects

- ESMF_Grid - representation of a grid
- ESMF_State - objects that hold gridded data
- ESMF_State consists of ESMF_Field, ESMF_Bundle, ESMF_Array
- Data is exchanged between ESMF Gridded components using ESMF_States



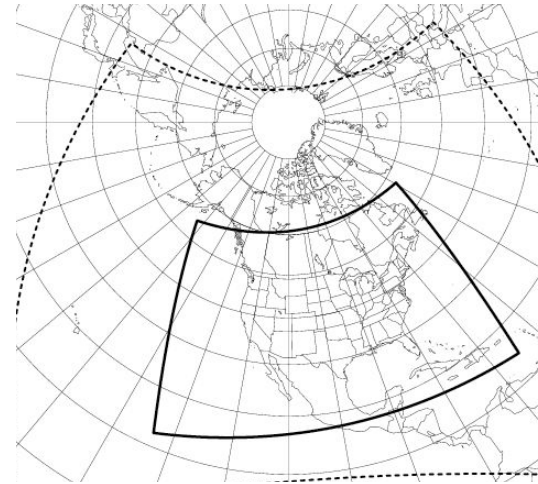
Key ESMF objects

- ESMF_Grid - representation of a grid
- ESMF_State - objects that hold gridded data
- ESMF_State consists of ESMF_Field, ESMF_Bundle, ESMF_Array
- Data is exchanged between ESMF Gridded components using ESMF_States



Key ESMF objects

- ESMF_Grid - representation of a grid
- ESMF_State - objects that hold gridded data
- ESMF_State consists of ESMF_Field, ESMF_Bundle, ESMF_Array
- Data is exchanged between ESMF Gridded components using ESMF_States



Running LIS

Rapid

A fatal exception 0E has occurred at 0028:C0011E36 in UXD UMM(01) + 00010E36. The current application will be terminated.

- * Press any key to terminate the current application.
- * Press CTRL+ALT+DEL again to restart your computer. You will lose any unsaved information in all applications.

Press any key to continue _

Running LIS

Need to specify the lis.config file that contains the list of configurable options

Need parameter data (static, dynamic)

Need forcing data

How to get started?

Run the canned testcases: (<http://lis.gsfc.nasa.gov/Source/testcases/>)

Find the lis.config file corresponding to each testcase under src/testcases

Generate parameter data at the desired resolution by using the LIS data processing programs

Software Architecture

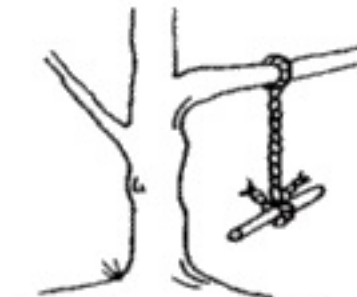


Copyright © 1995 United Feature Syndicate, Inc.
Redistribution in whole or in part prohibited

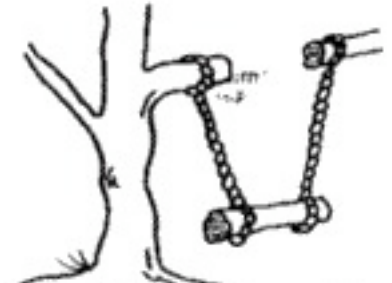
© Dave Mocko, Matt Garcia, Charles Alonge

Software Design

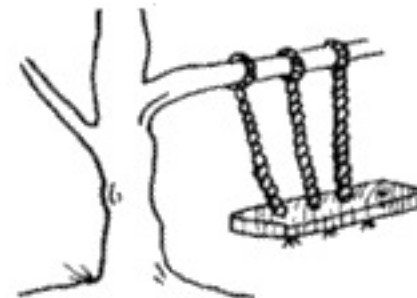
- Paradigm of “Problem Solving Environments” or “Expert Systems”
- an integrated system provides the expert tools for complex domains
- LIS is a PSE for hydrologic modeling applications
- LIS is designed as an object-oriented framework



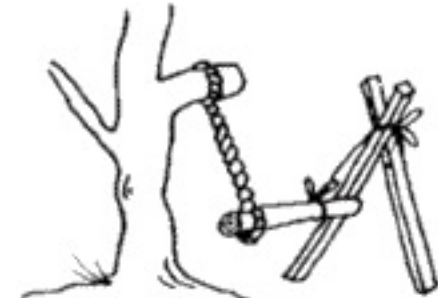
What the user asked for



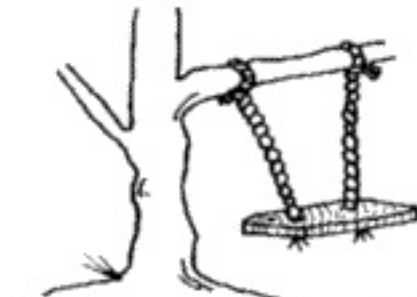
How the analyst saw it



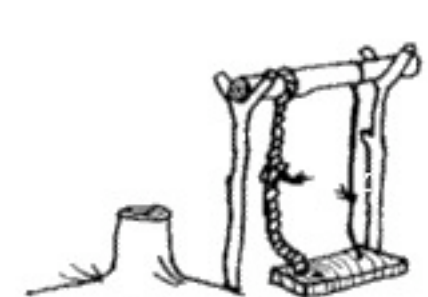
How the system was designed



As the programmer wrote it



What the user really wanted



How it actually works

Object Oriented Programming



Think Objects



Modularity : Source code for an object, written and maintained independent of the source code for other objects



Reusability: if the object already exists, you can use that object in your application



Extensibility: Can be customized for new applications



Inversion of Control - “Don’t call us, we’ll call you”

Generic code controls execution of problem-specific code



States and Behavior

Gear

Speed

Pedal stance

change gear

apply brakes

speed up

Object Oriented Programming



Think Objects



Modularity : Source code for an object, written and maintained independent of the source code for other objects



Reusability: if the object already exists, you can use that object in your application



Extensibility: Can be customized for new applications



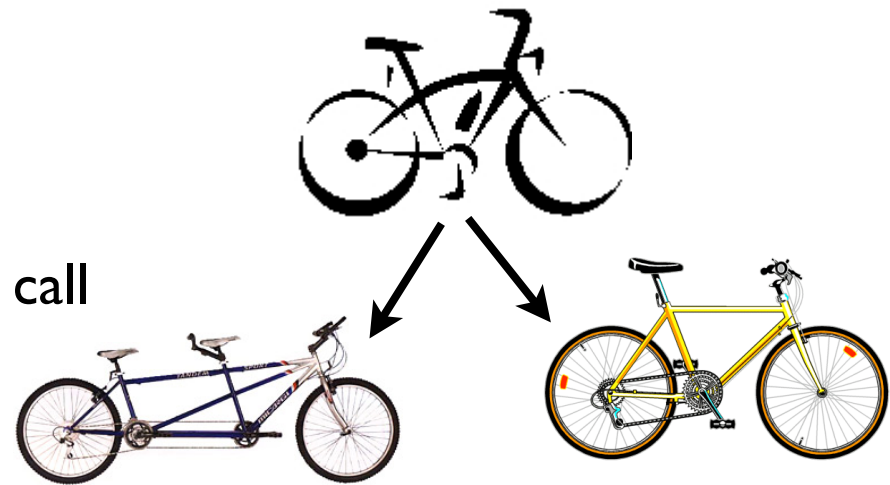
Inversion of Control - “Don’t call us, we’ll call you”

Generic code controls execution of problem-specific code

Object Oriented Programming

- Think Objects
- Modularity : Source code for an object, written and maintained independent of the source code for other objects
- Reusability: if the object already exists, you can use that object in your application
- Extensibility: Can be customized for new applications
- Inversion of Control - “Don’t call us, we’ll call you”

Generic code controls execution of problem-specific code



Object Oriented Programming



Think Objects



Modularity : Source code for an object, written and maintained independent of the source code for other objects



Reusability: if the object already exists, you can use that object in your application



Extensibility: Can be customized for new applications



Inversion of Control - “Don’t call us, we’ll call you”

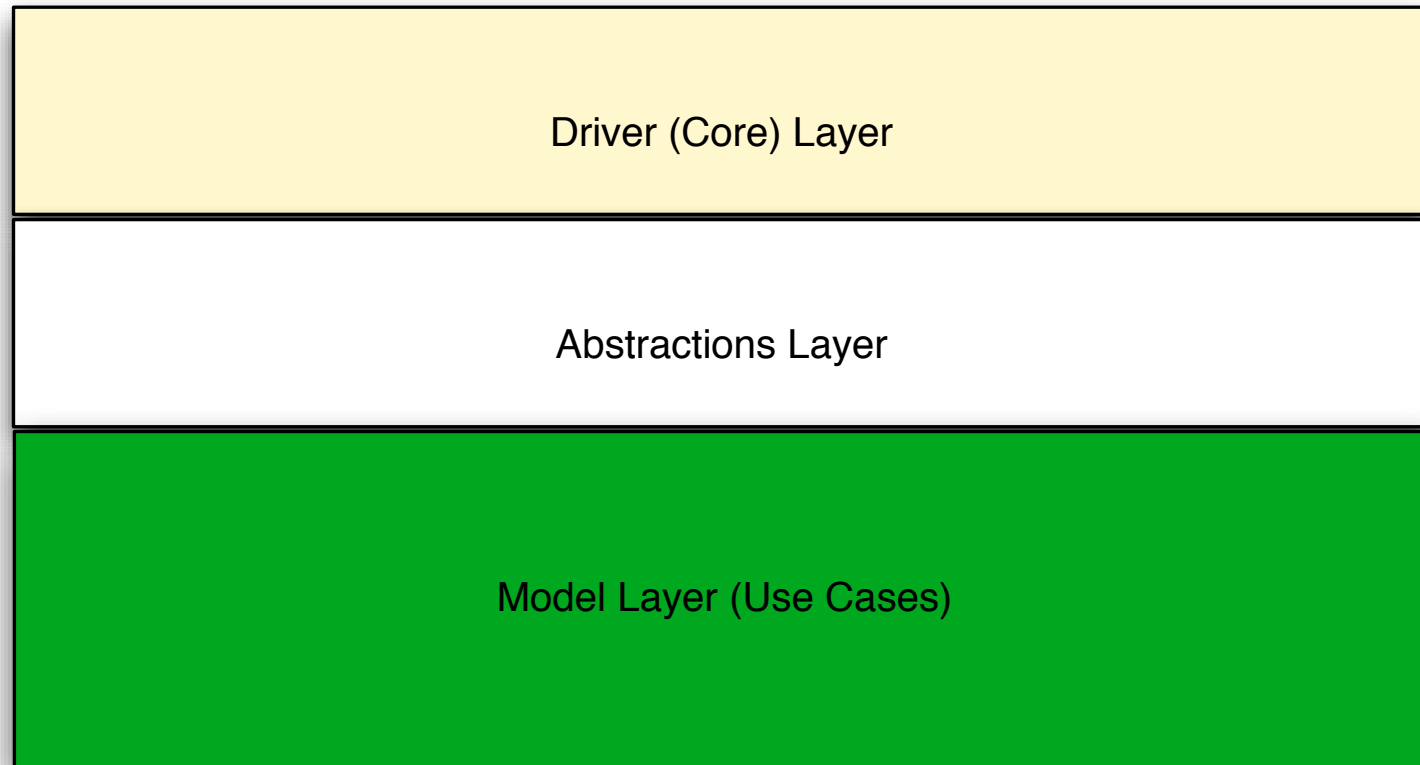
Generic code controls execution of problem-specific code

Object Oriented Programming

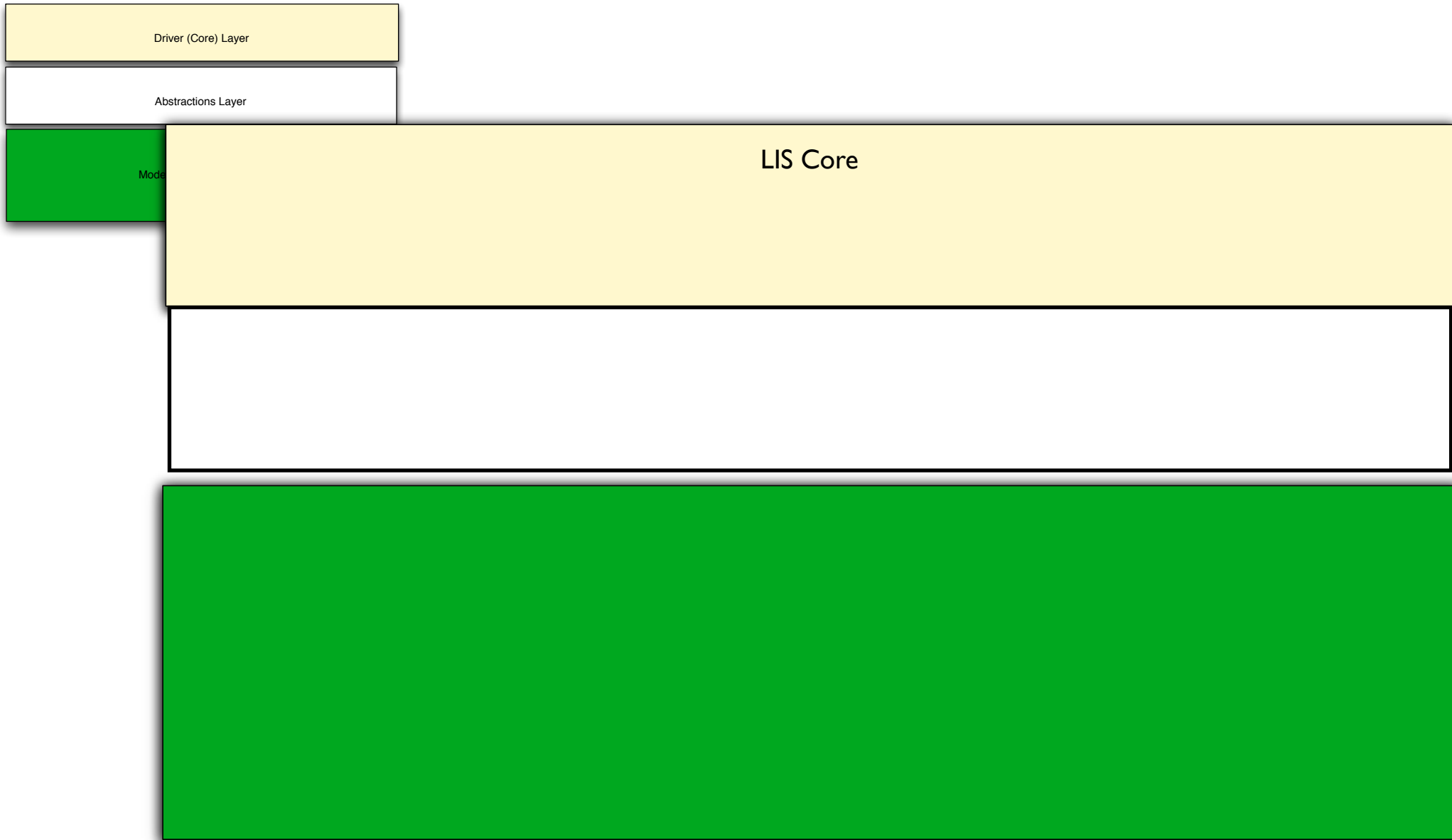
- 🔩 Think Objects
- 🔩 Modularity : Source code for an object, written and maintained independent of the source code for other objects
- 🔩 Reusability: if the object already exists, you can use that object in your application
- 🔩 Extensibility: Can be customized for new applications
- 🔩 Inversion of Control - “Don’t call us, we’ll call you”

Generic code controls execution of problem-specific code

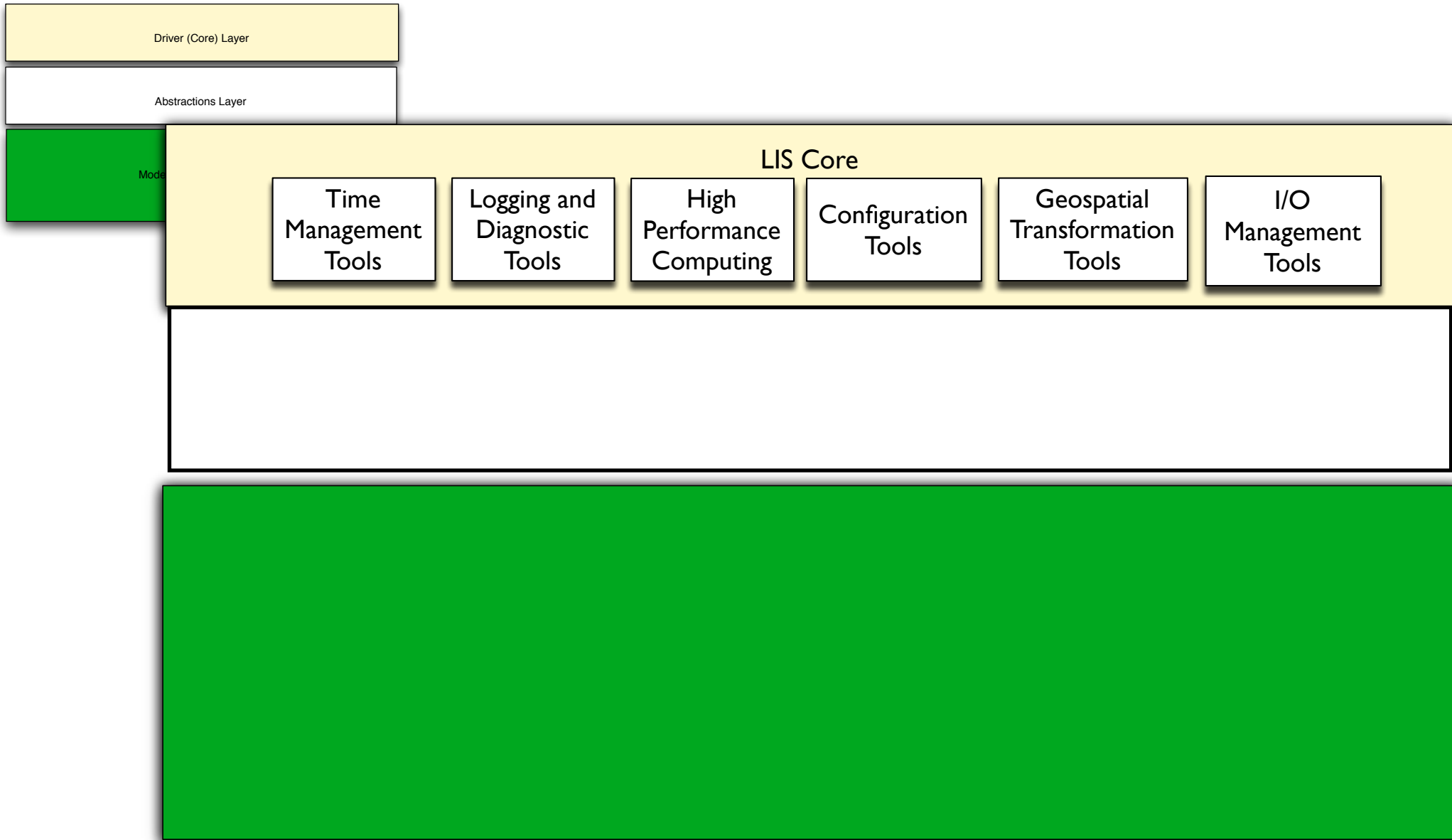




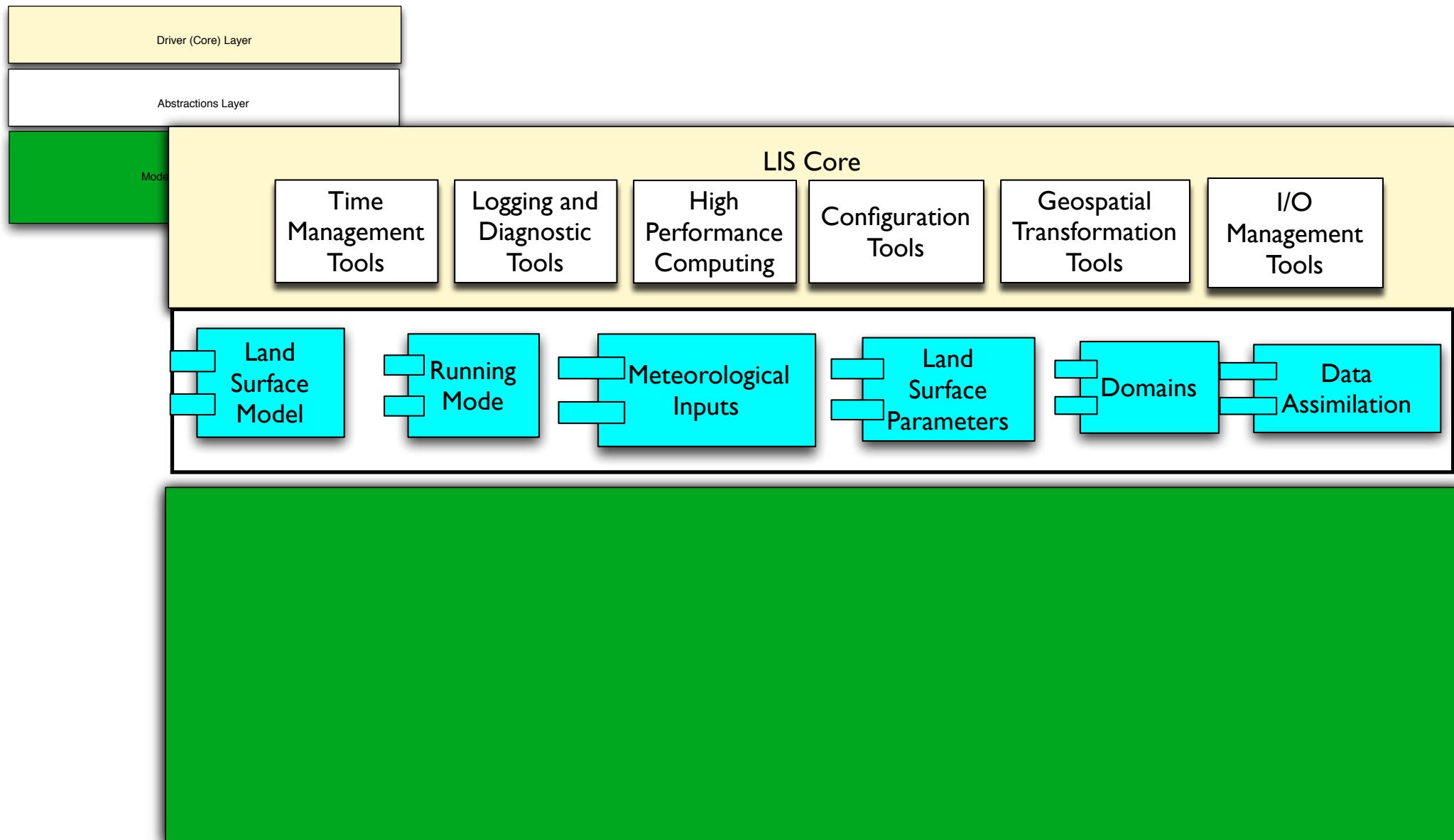
LIS software architecture



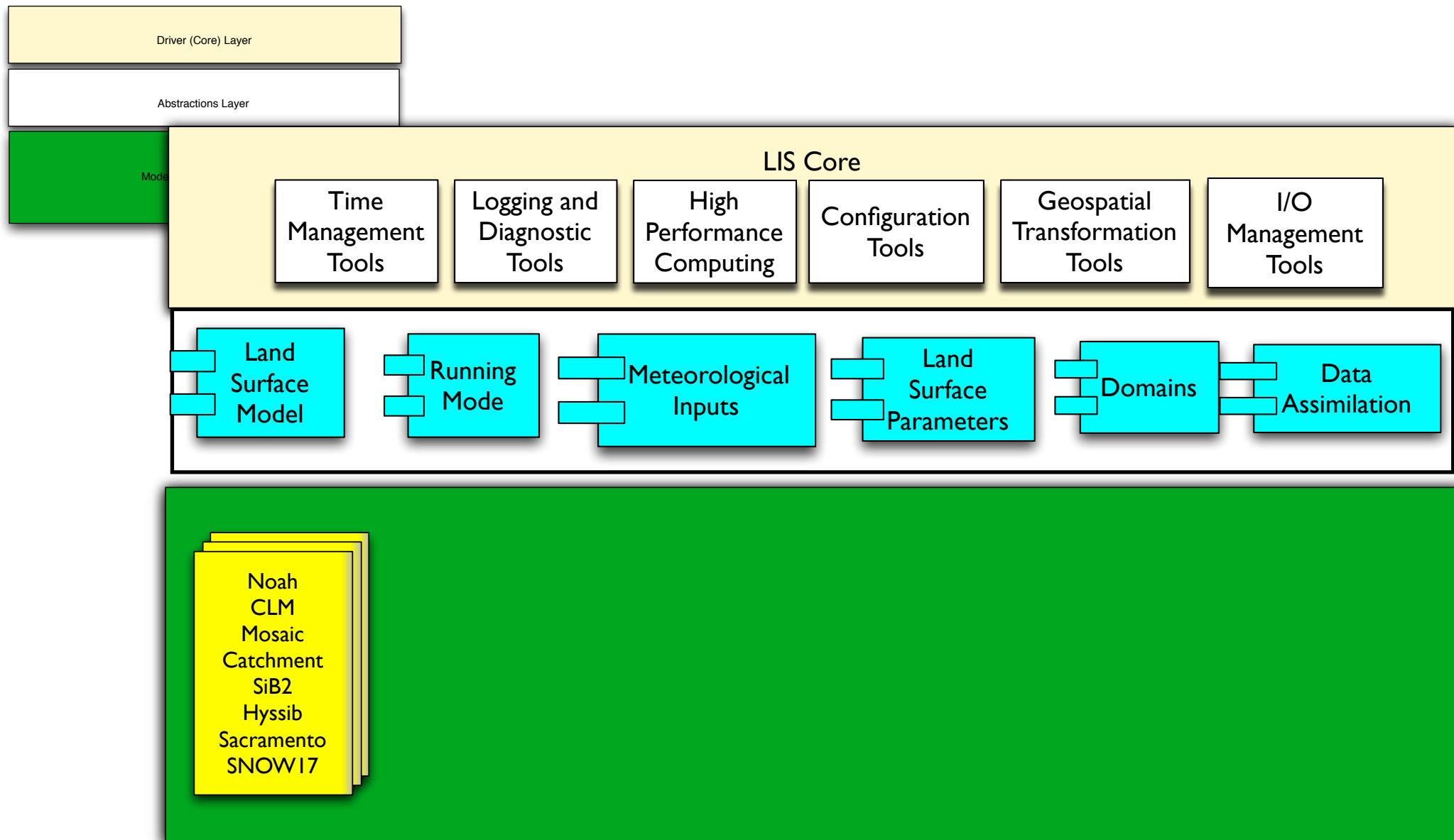
LIS software architecture



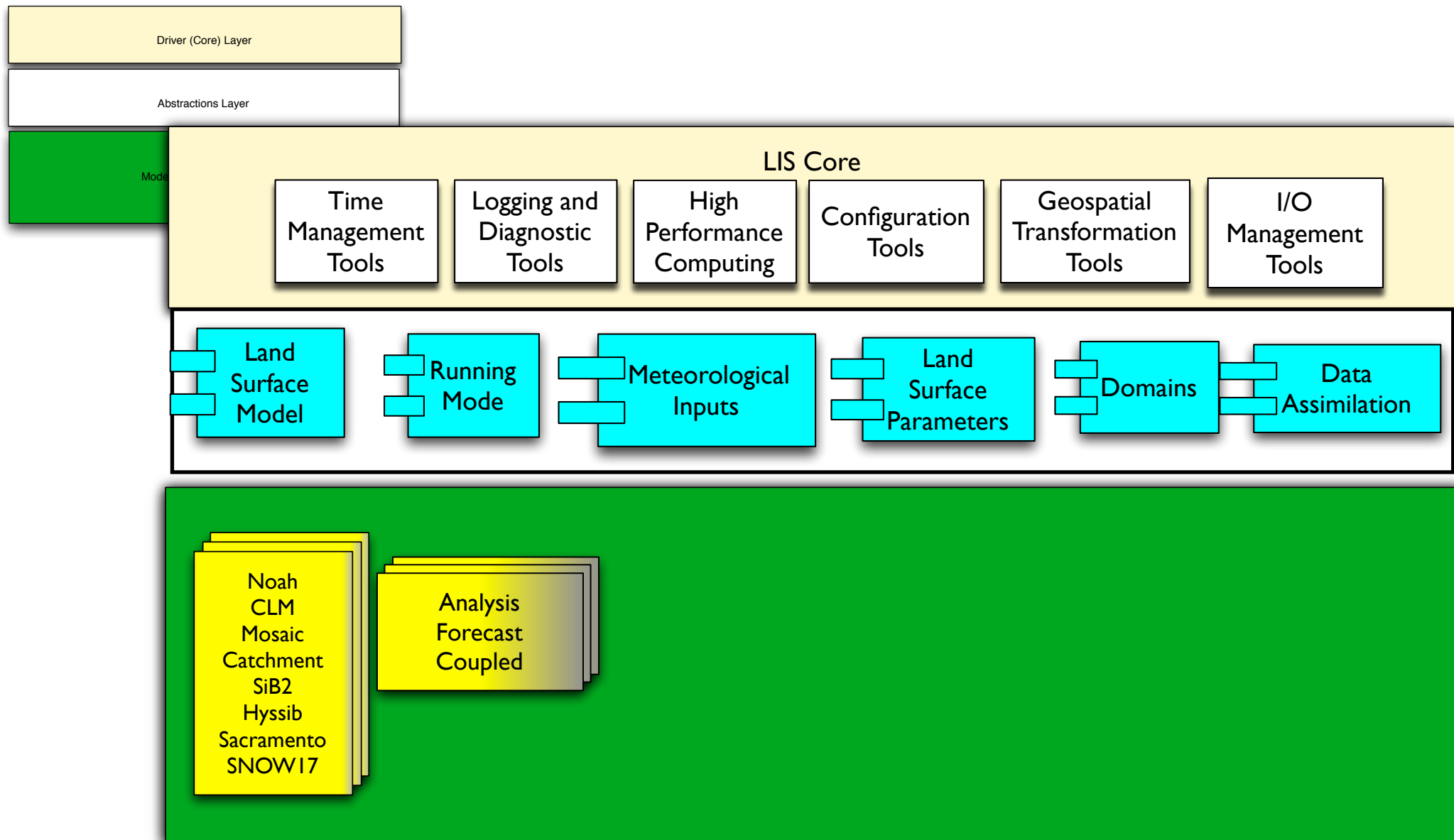
LIS software architecture



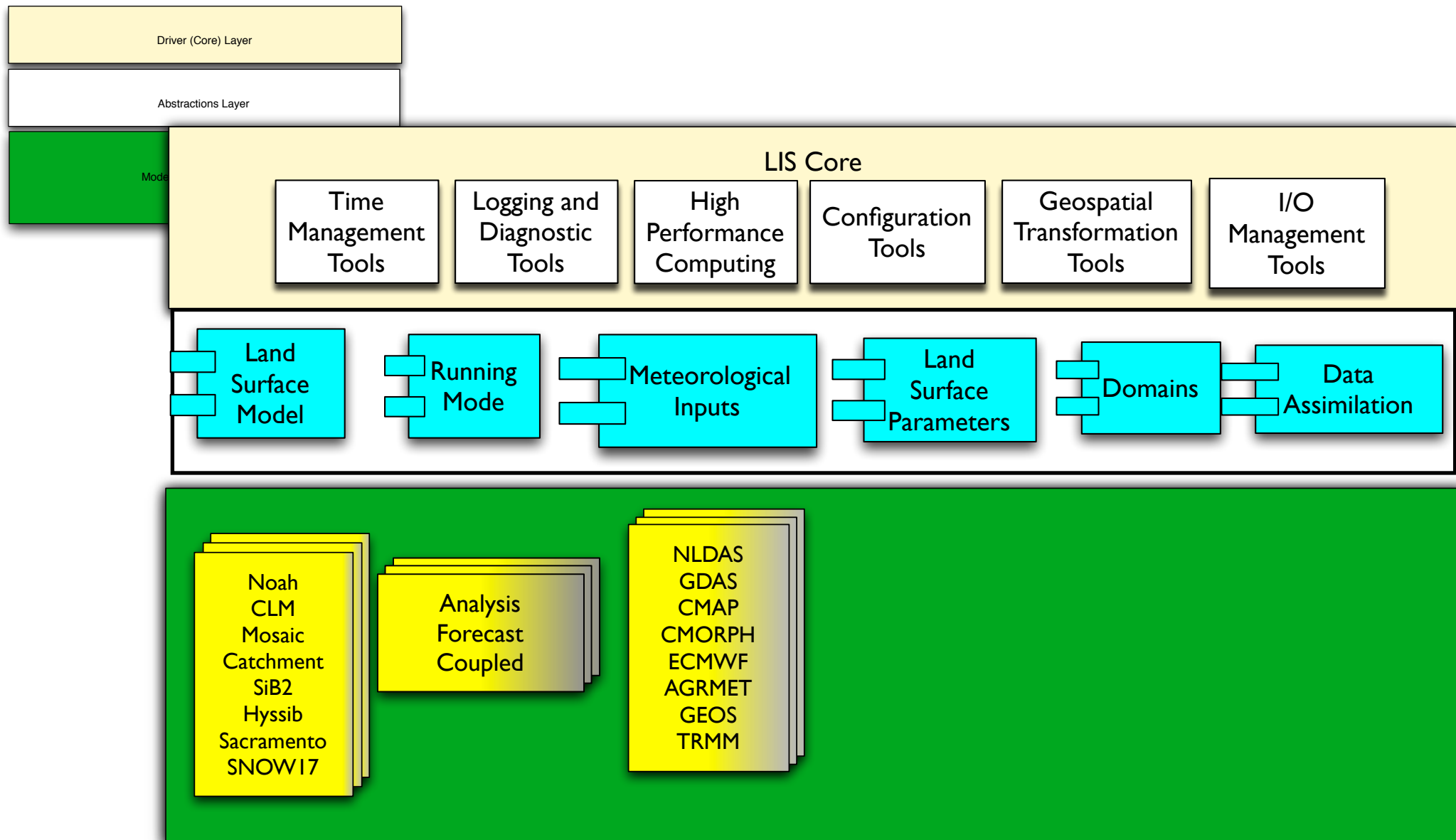
LIS software architecture



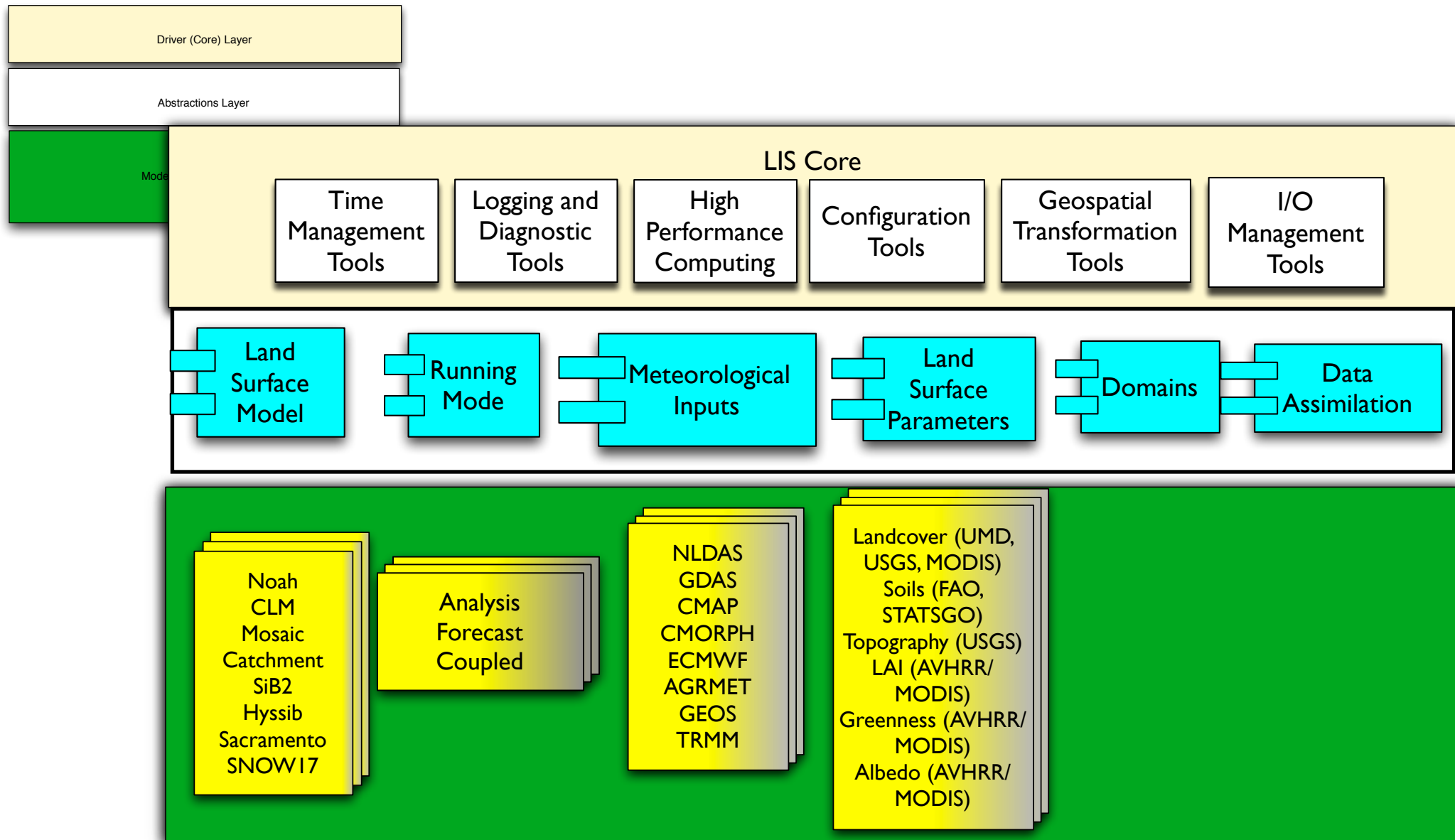
LIS software architecture



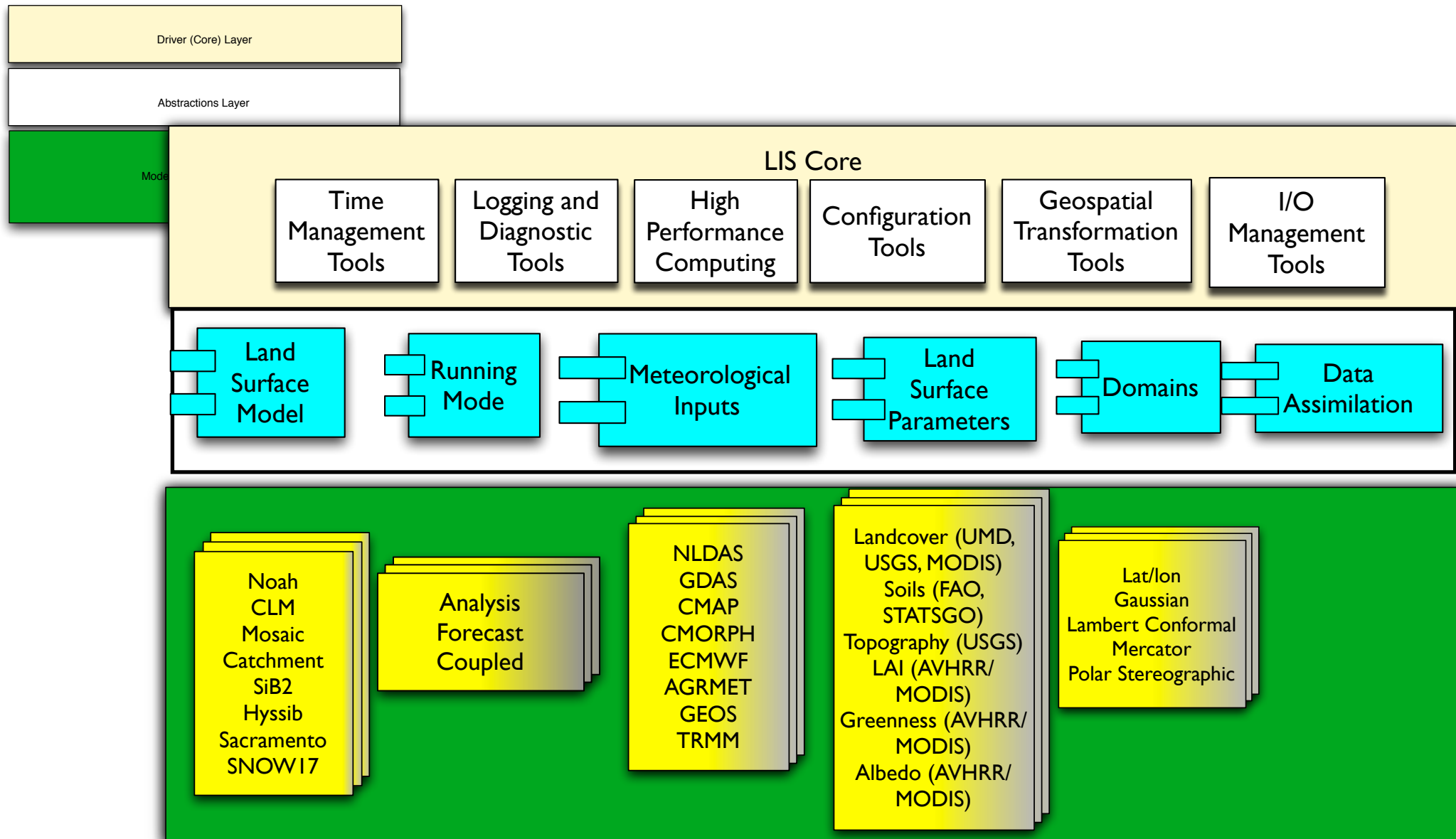
LIS software architecture



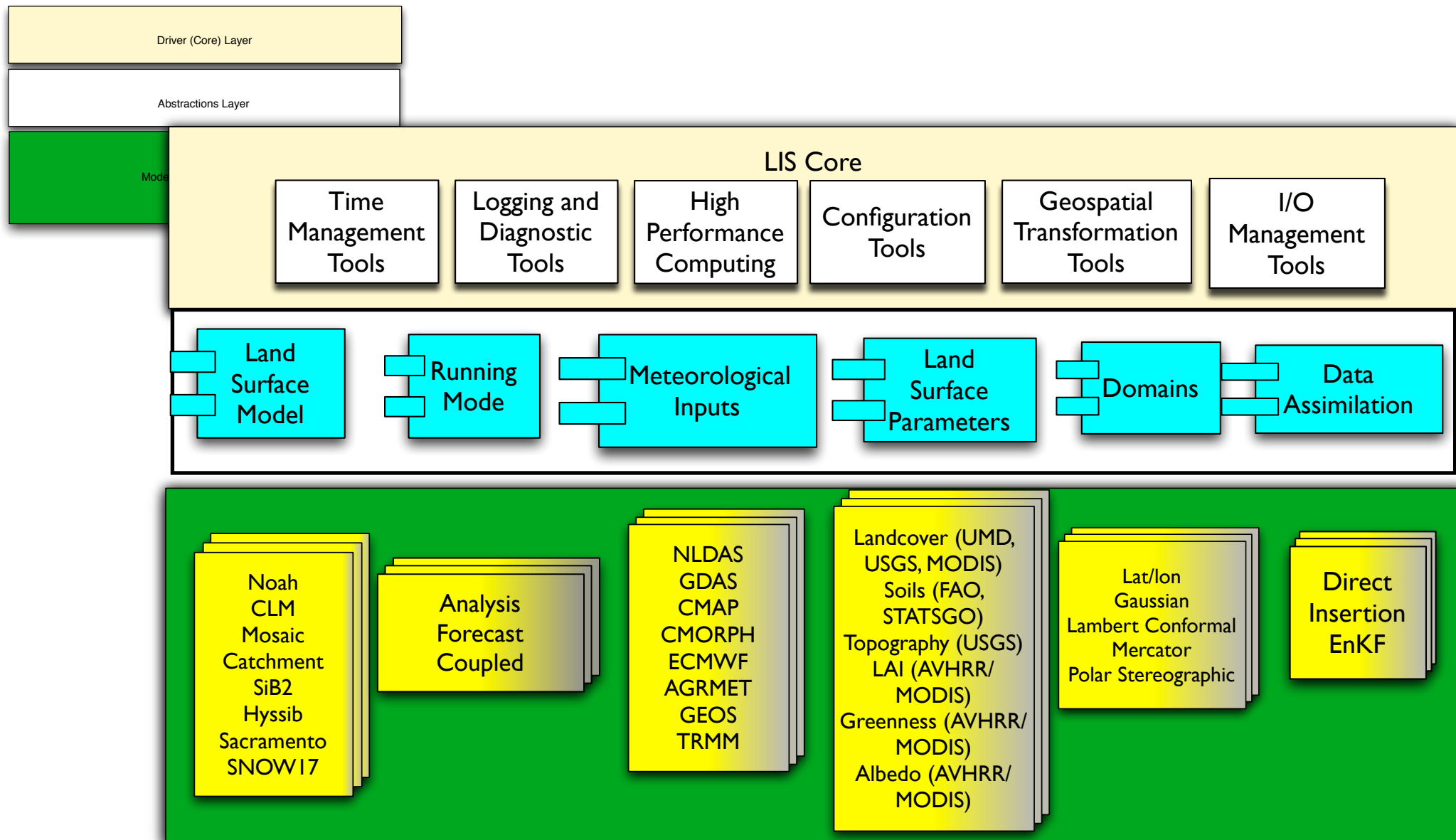
LIS software architecture



LIS software architecture



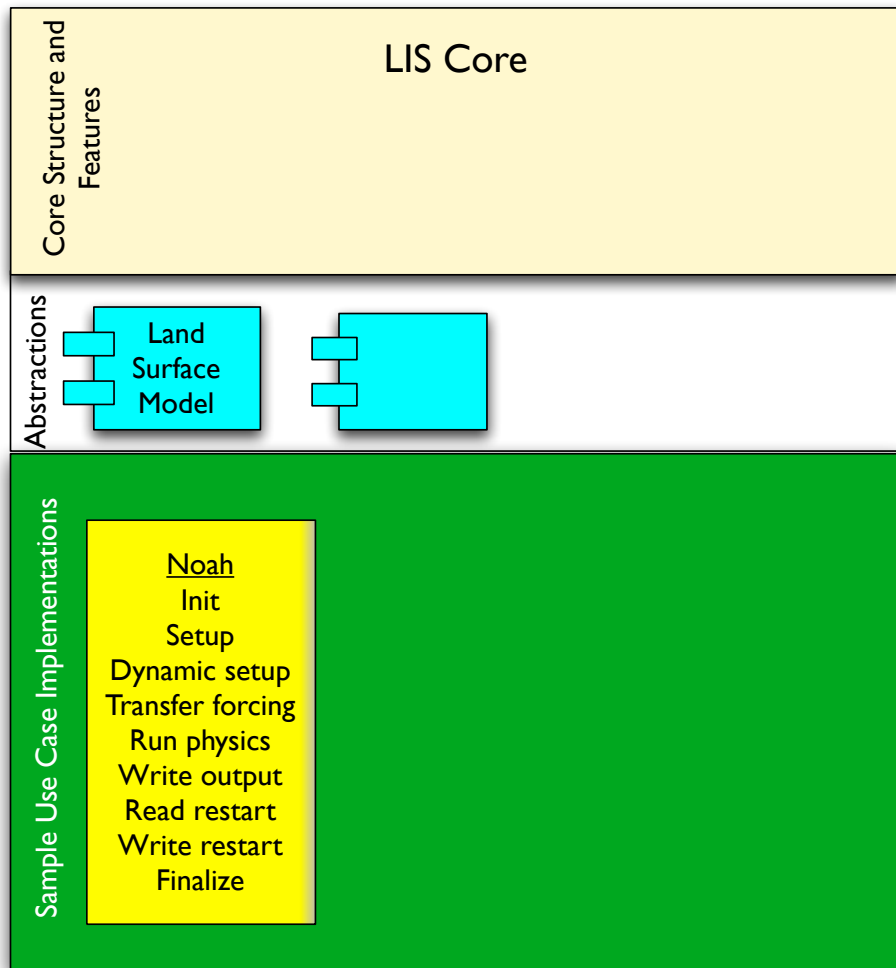
LIS software architecture



LIS software architecture

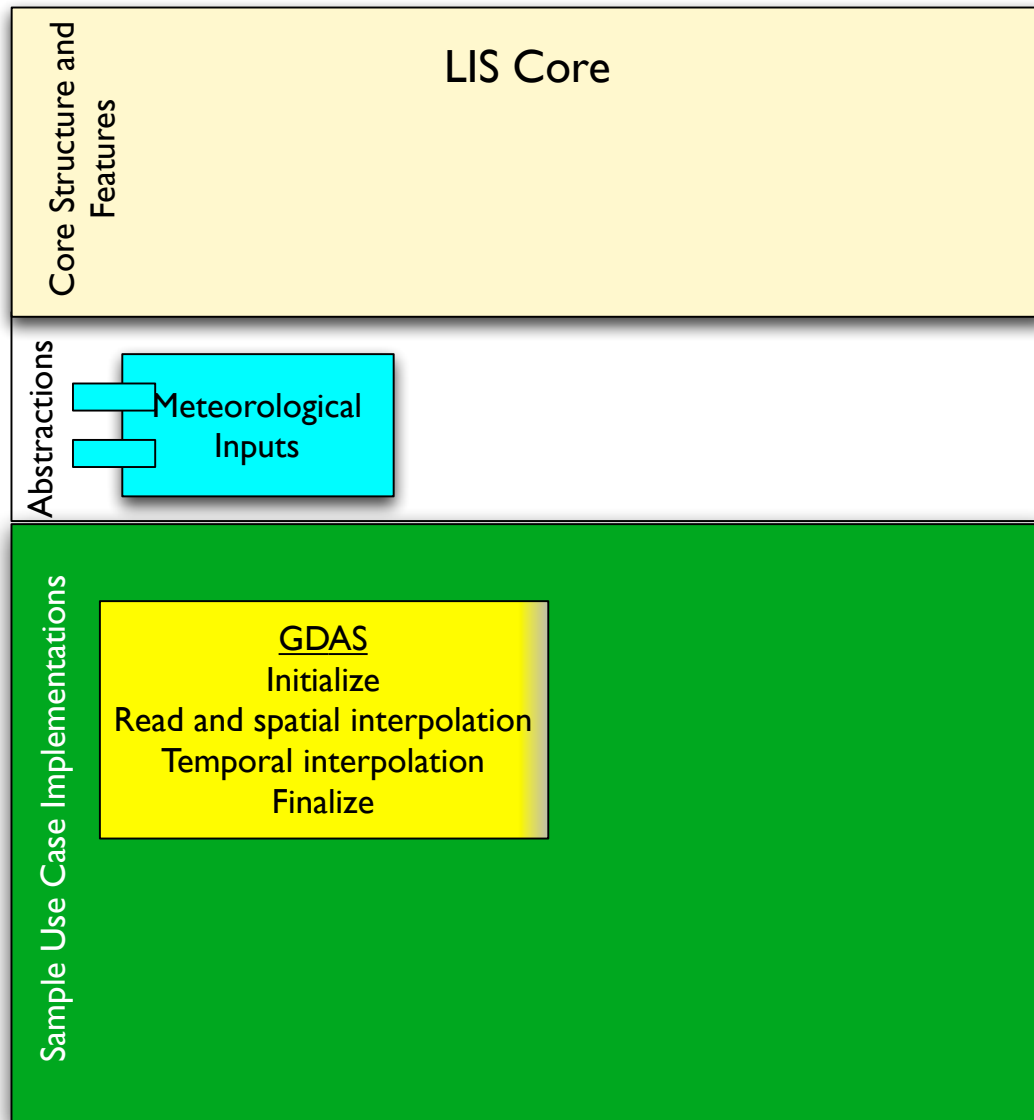
Customizing LIS


How do we add a new LSM?



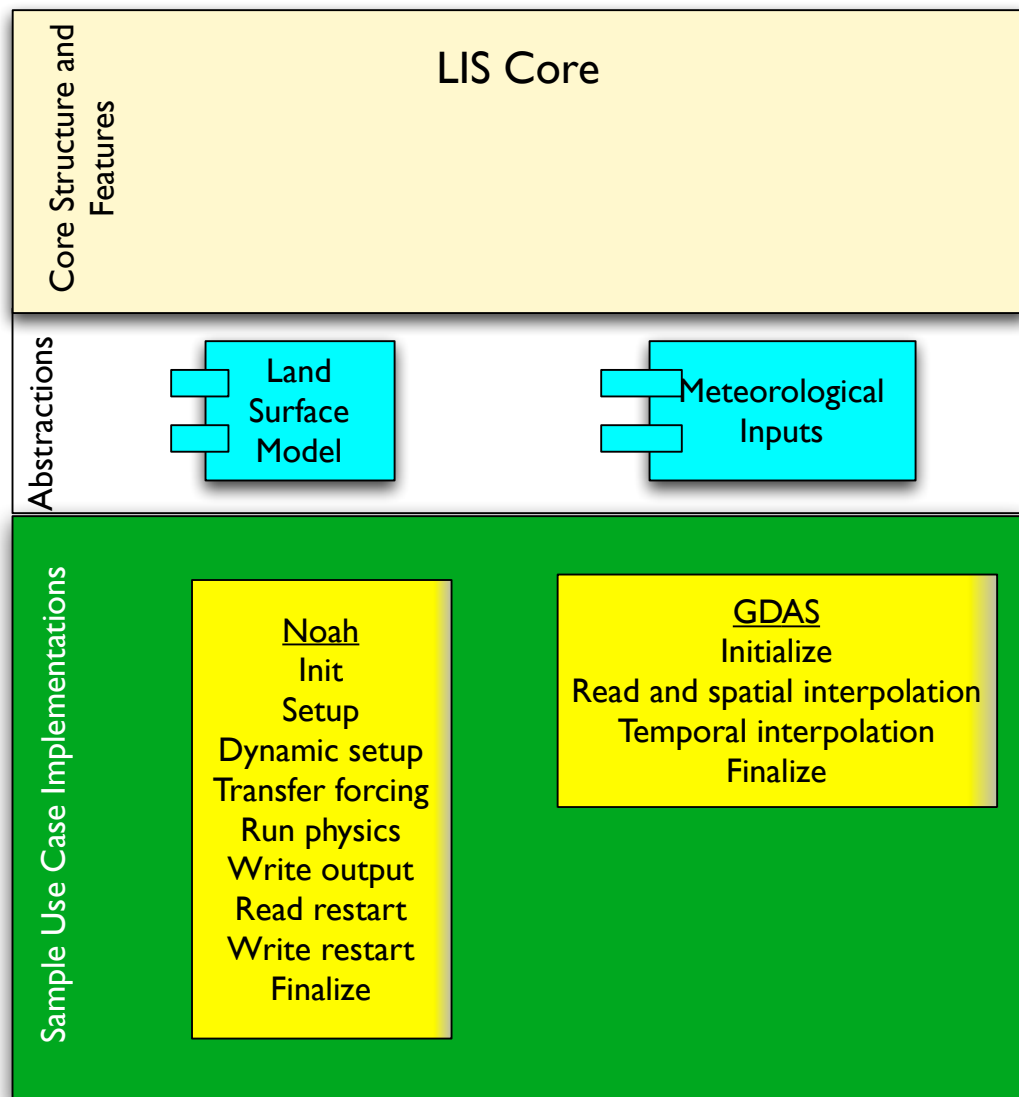
- Need to implement a set of interfaces related to the operation of a land surface model
- In LIS, these abstract implementations are known as “plugins”
- under `src/plugins`

How do we add a new forcing scheme?



 Extend the abstract interfaces related to a forcing scheme

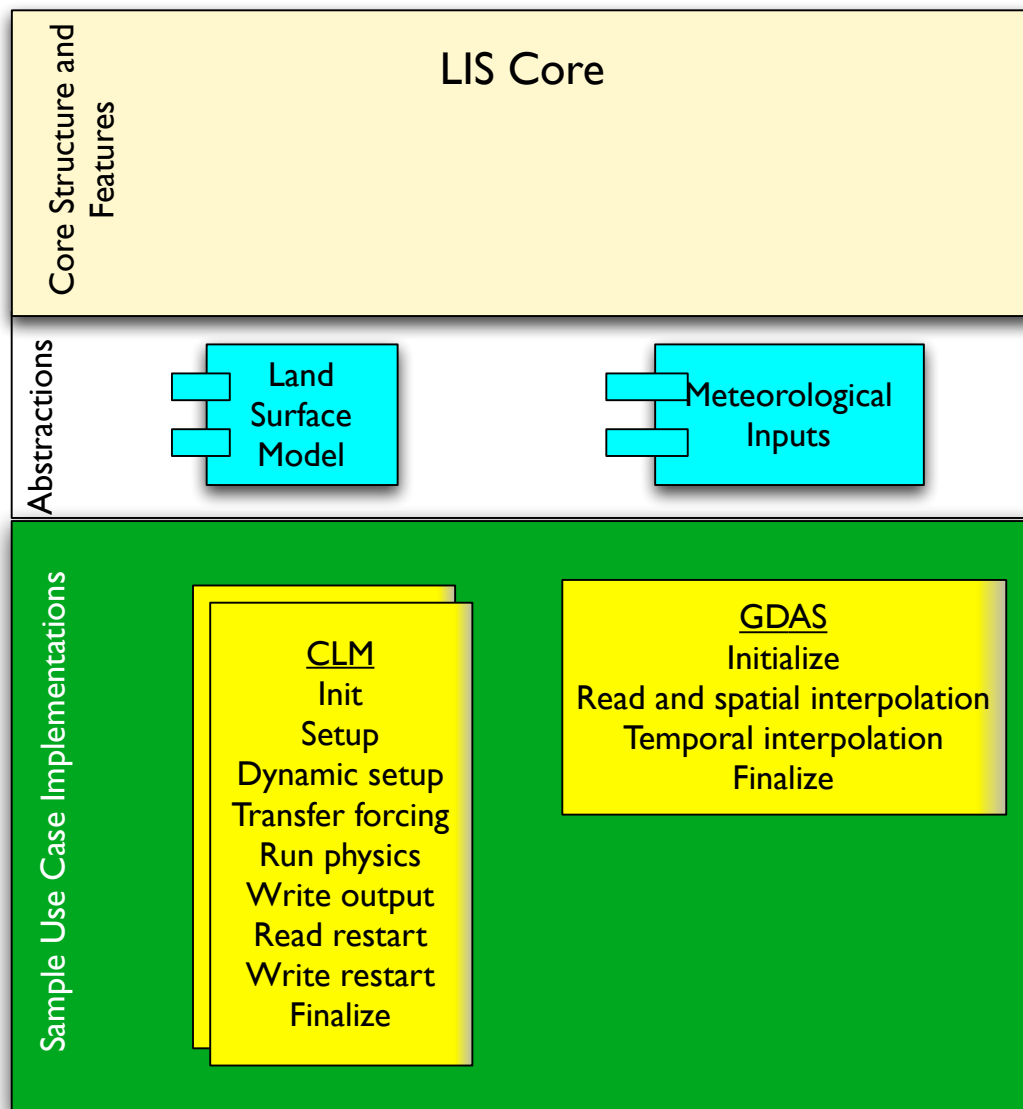
Combining these components



LIS provides the “wirings” between the abstract implementations

Incorporating these components through plugins automatically ensure their integrated and interoperable use

Combining these components

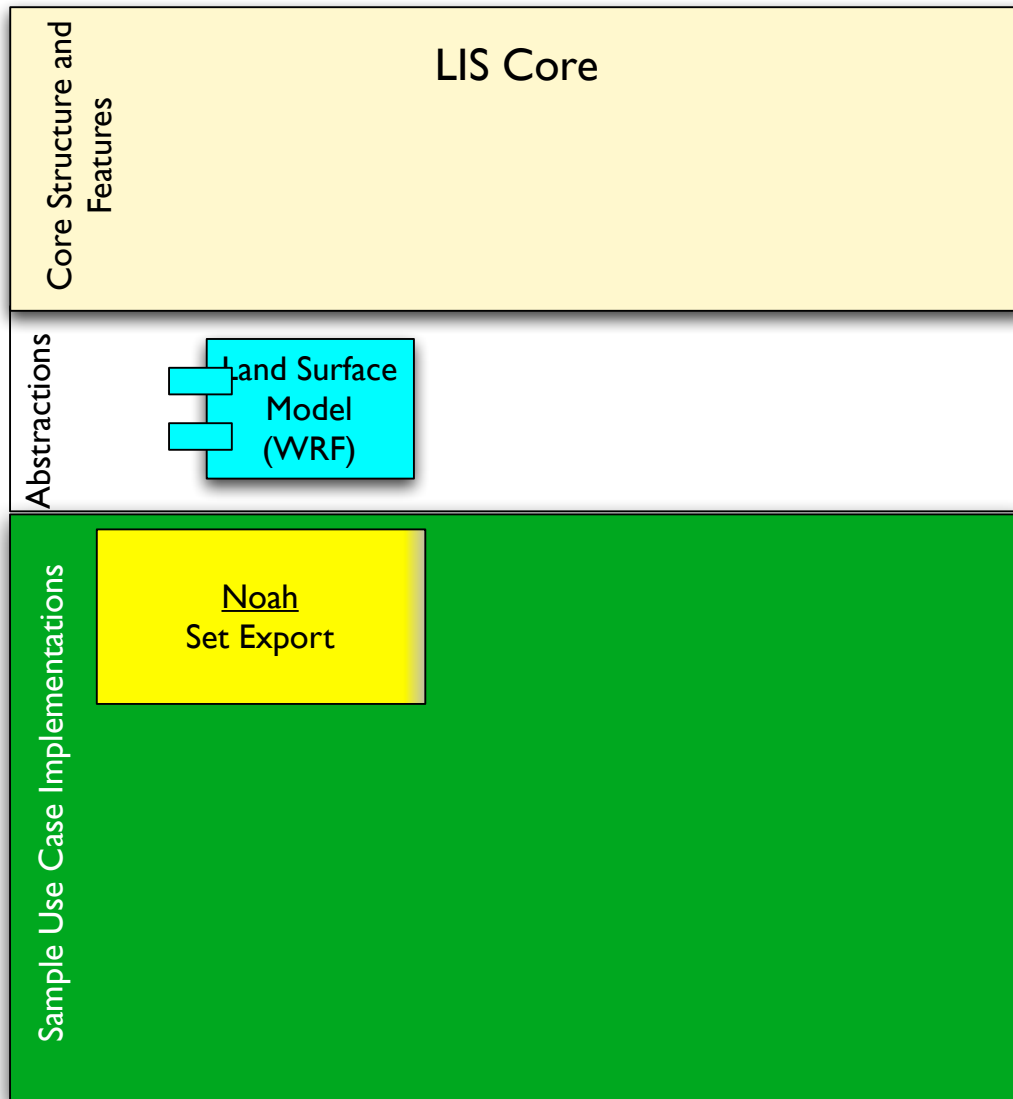


LIS provides the “wirings” between the abstract implementations

Incorporating these components through plugins automatically ensure their integrated and interoperable use

Coupling to other earth system models (WRF)

Enable coupling to WRF

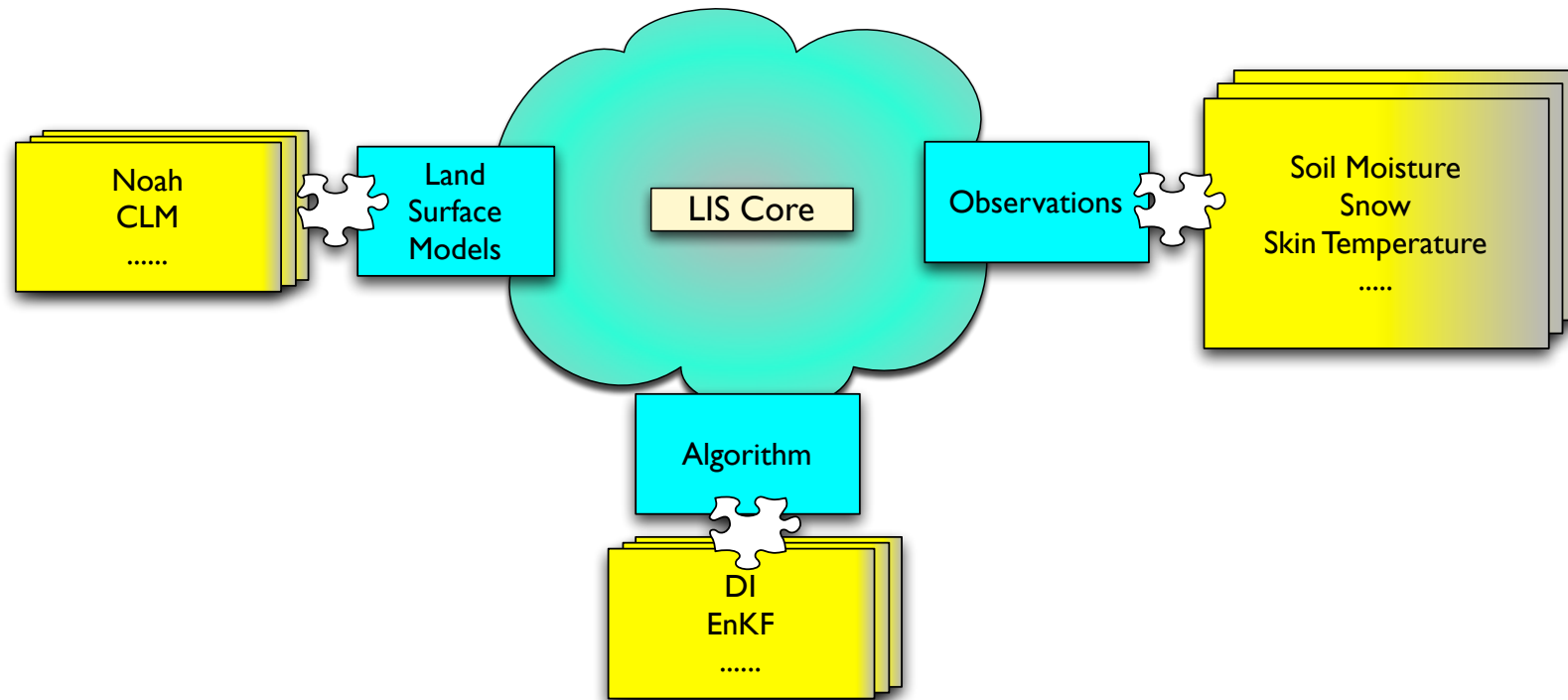


- ✓ A routine that specifies a set of export states to WRF needs to be defined
- ✓ The import state to LIS is a standard list of forcing, common to all LSMs

Kumar et al. (2007): An integrated high-resolution hydrometeorological modeling testbed using LIS and WRF, Environmental Modeling and Software, 23 (2), 169-181.

Data Assimilation

Abstractions related to Data Assimilation

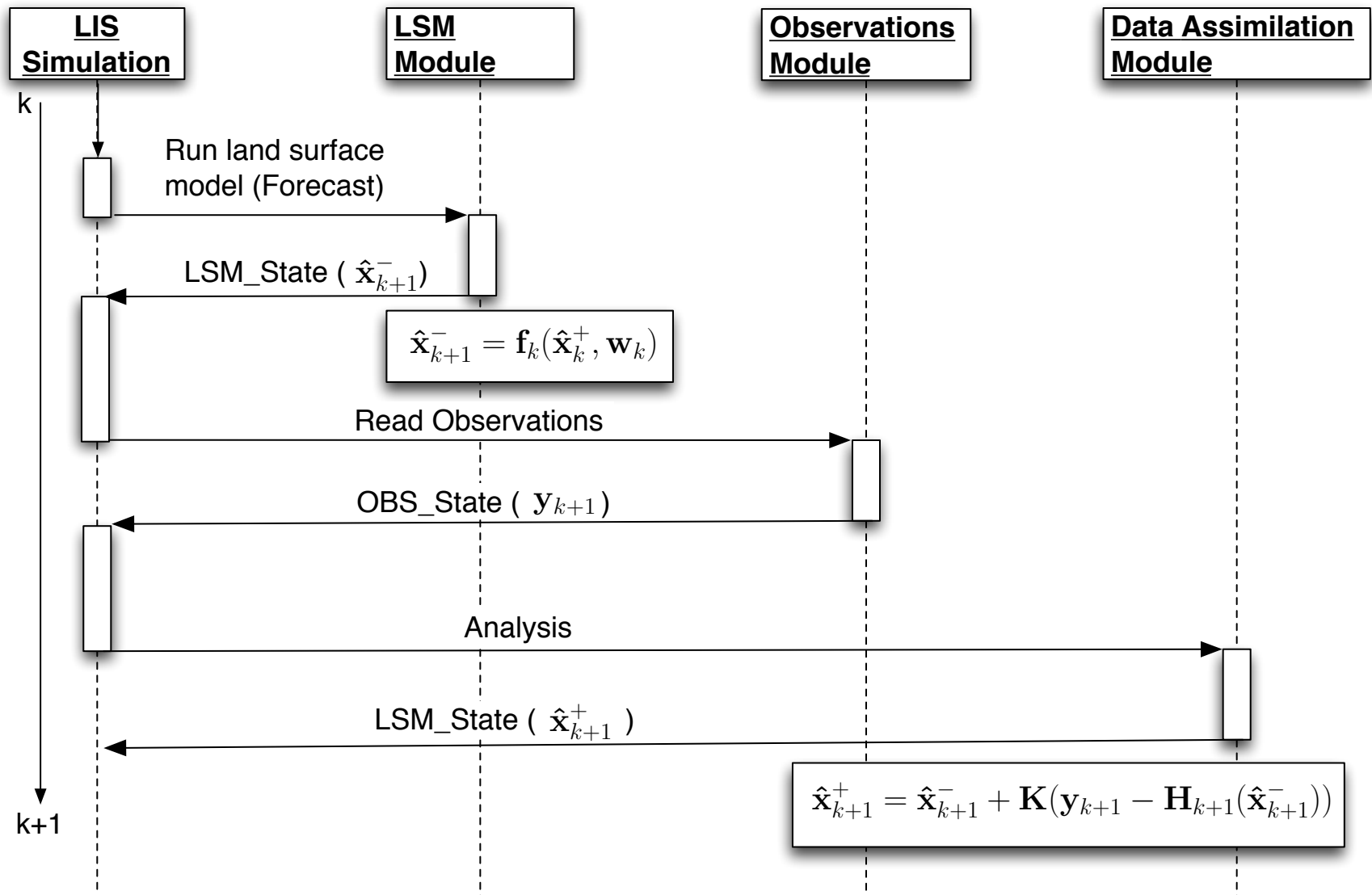


Goal:

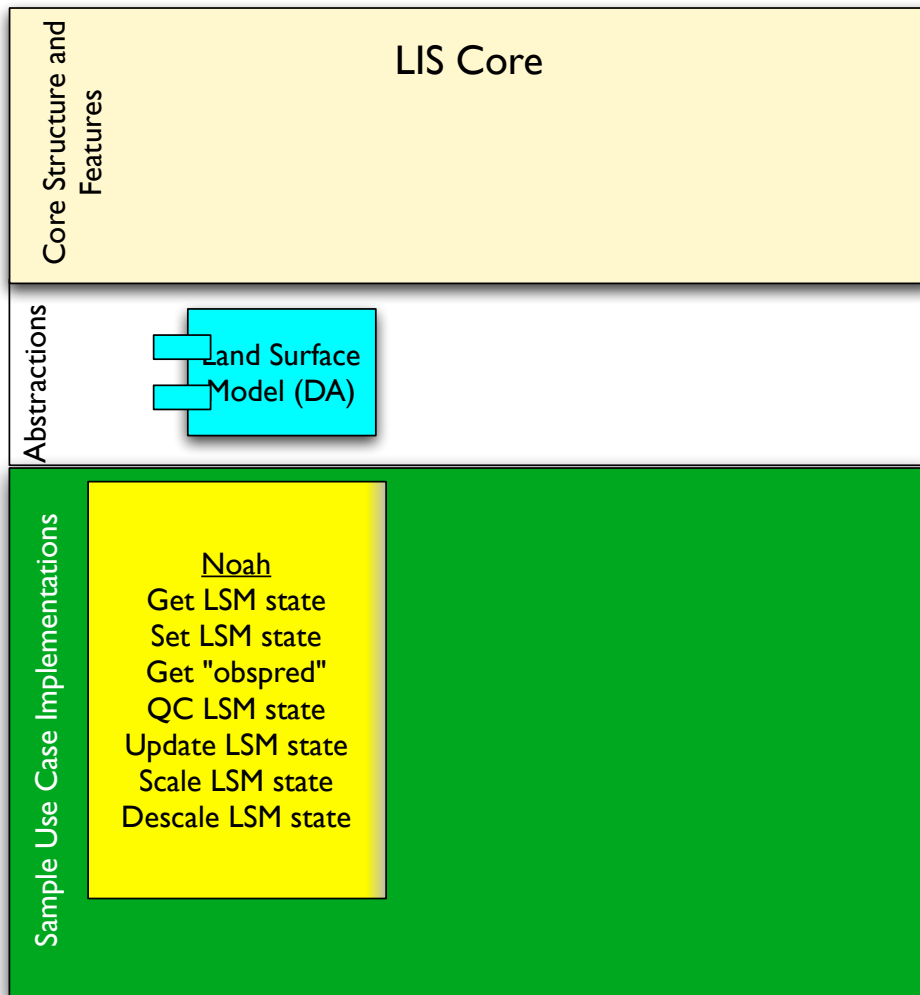
Interoperability: Once you define an observation plugin, it should work with an existing assimilation algorithm and LSM

Kumar et al. (2008): A Land Surface Data Assimilation Framework using the Land Information System: Description and Application, *Advances in Water Resources*, doi:10.1016/j.advwatres.2008.01.013.

Sequence of component interactions



How do we add a new DA instance?



Step I



Add DA related plugins for the LSM

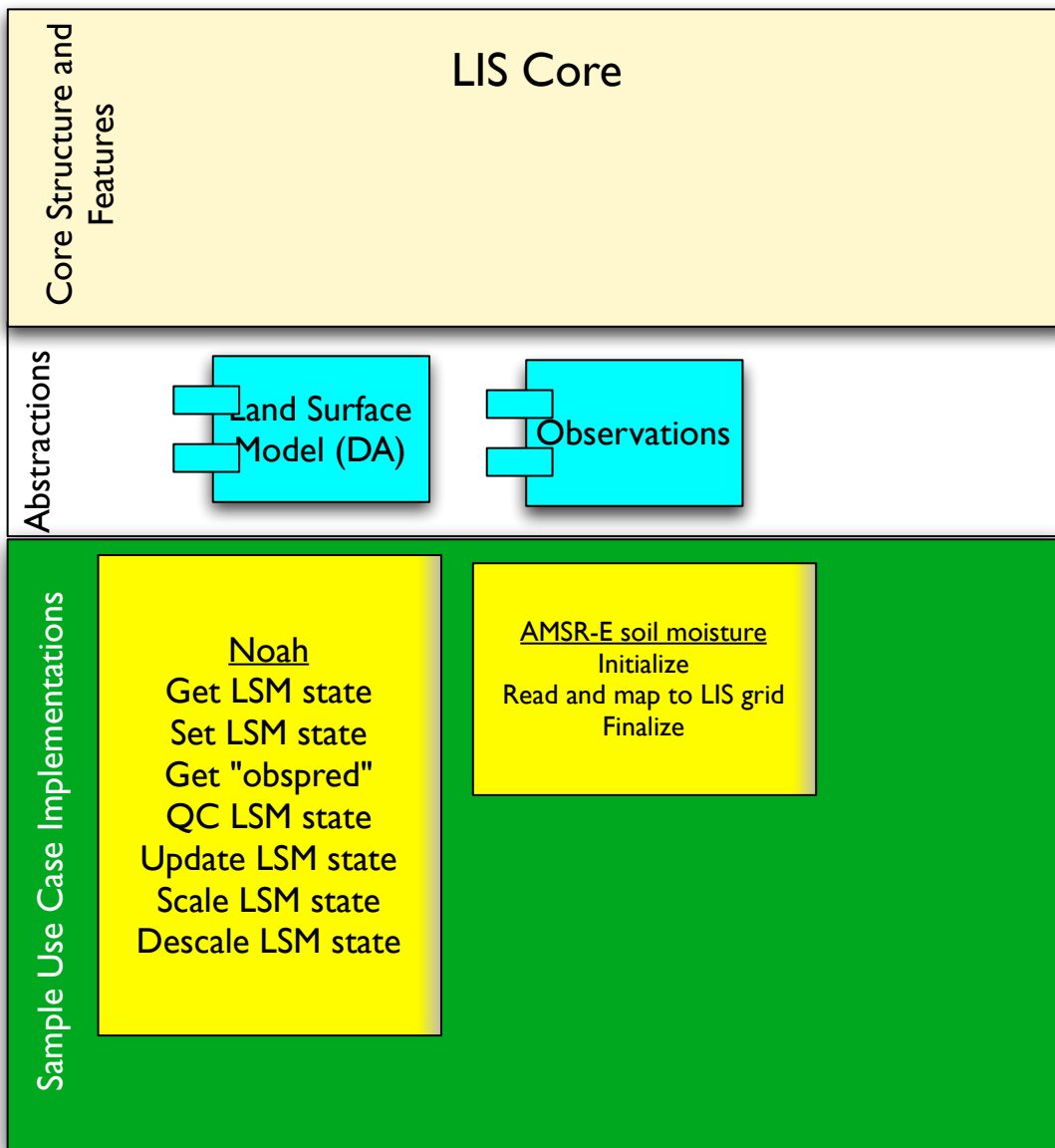


Identify LSM prognostic variables



Define the "obspred" - what the model thinks the observation should be

How do we add a new DA instance? (contd.)

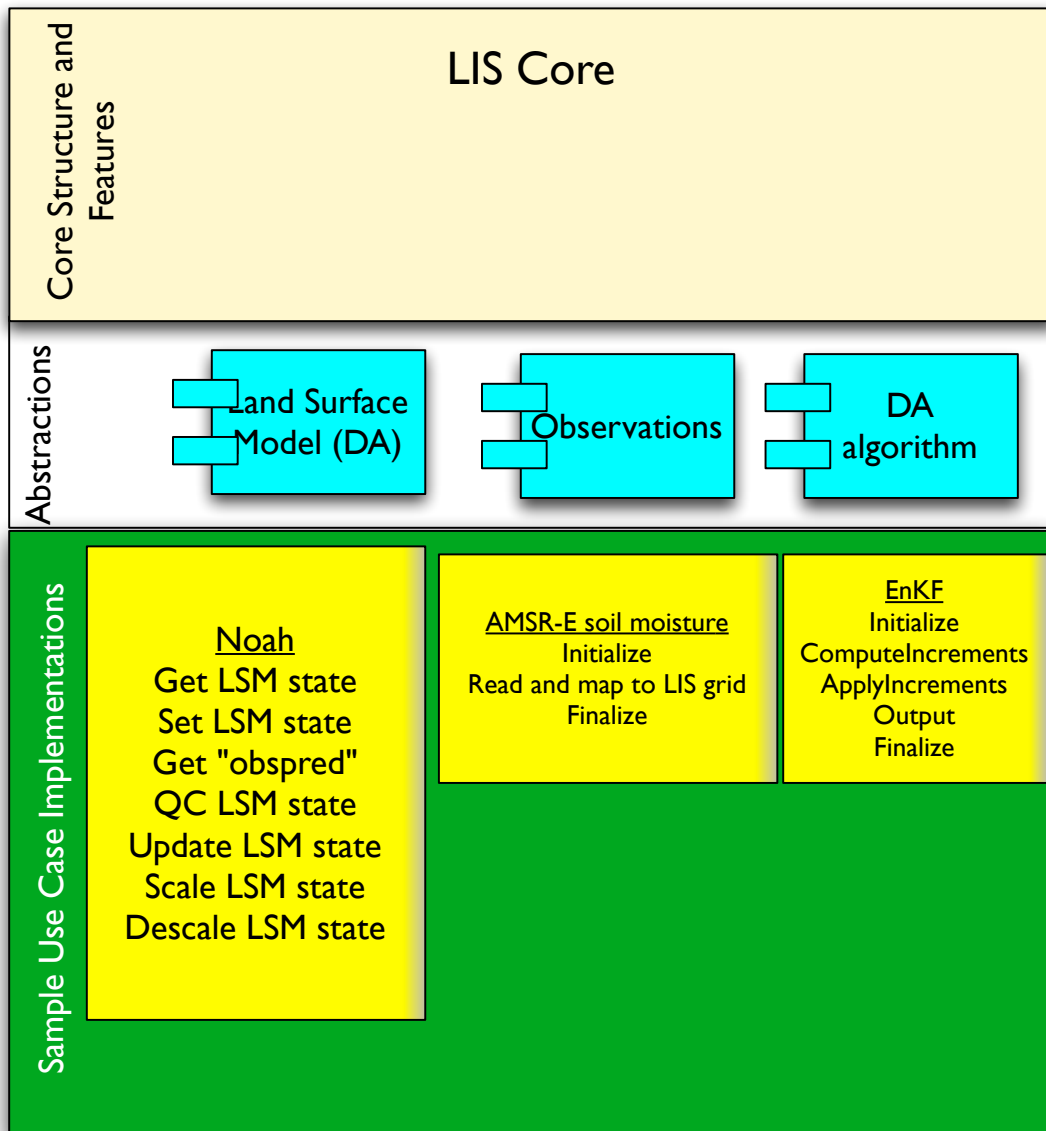


Step2



Add DA related plugins for observation source

How do we add a new DA instance? (contd.)



Step3



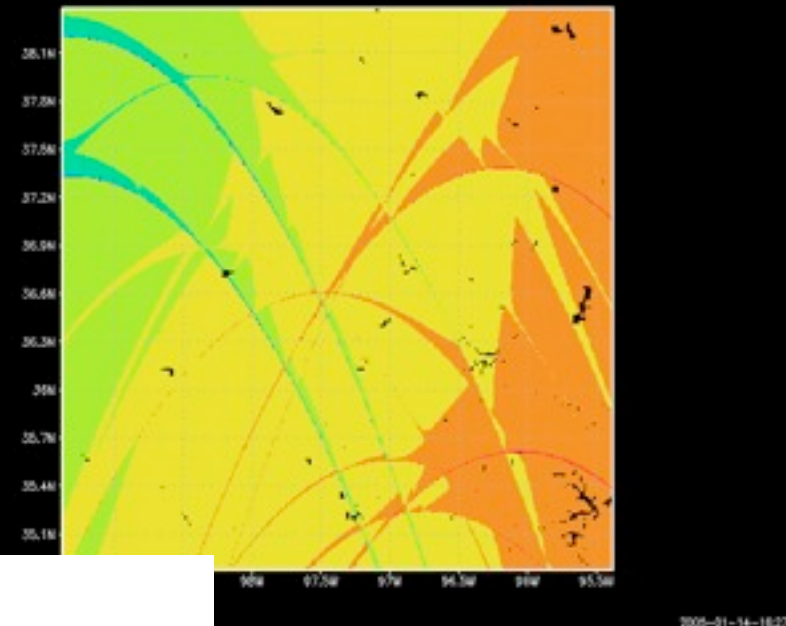
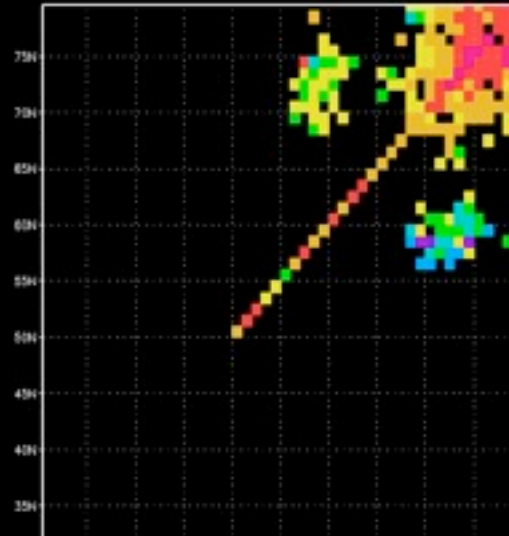
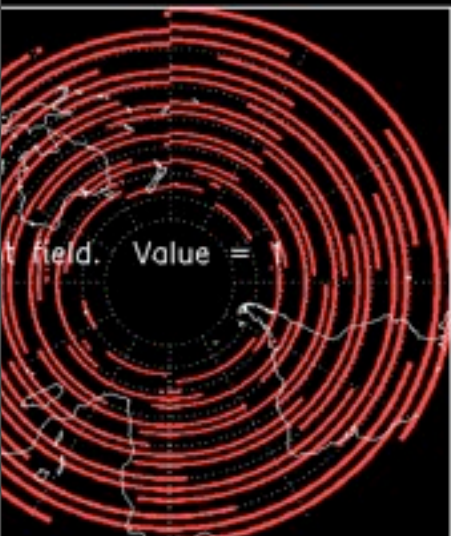
Add DA algorithm related plugins



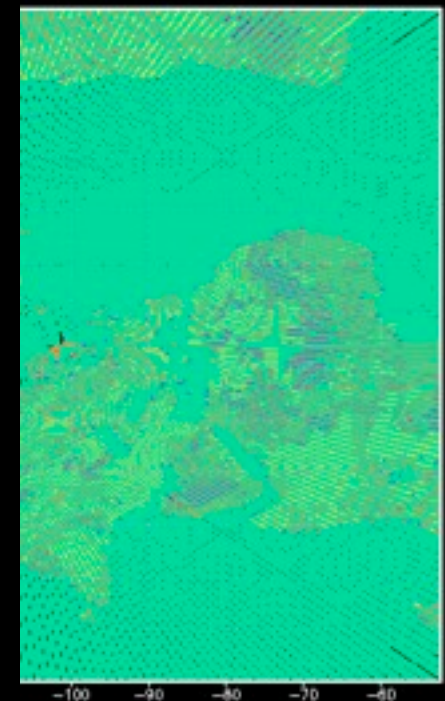
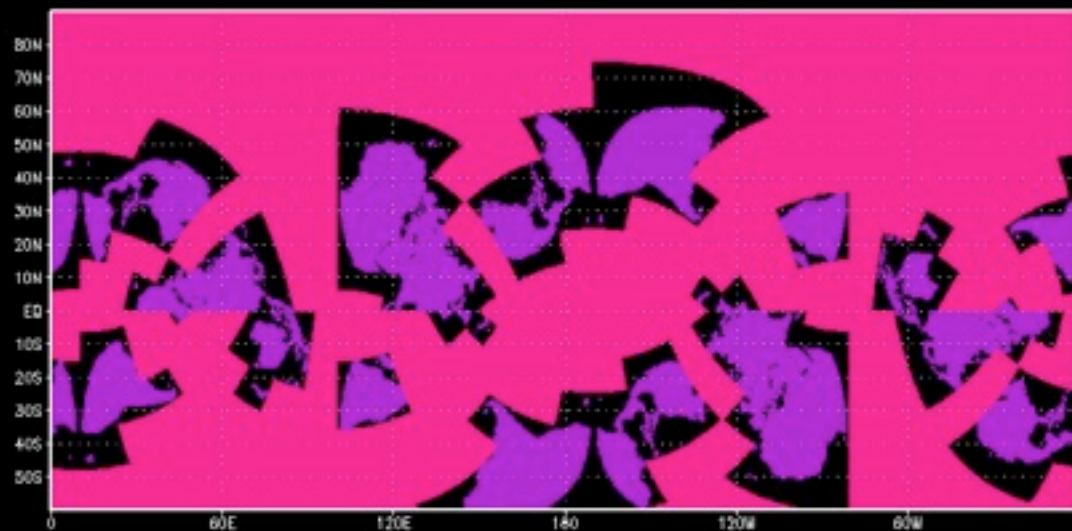
Users can simply reuse the existing implementation of EnKF and direct Insertion



The wirings between these implementations are automatically done by virtue of the connections between the abstract implementations



LIS output



LIS output visualization

- 📌 Binary/Grib I/NETCDF output
- 📌 Use GrADS, IDL, etc.
- 📌 A few utilities: src/utls/
 - 📌 src/utls/grads - program to generate a LIS control file
 - 📌 src/utls/ensemble - program to generate a LIS ensemble restart file from a single member integration

Questions?